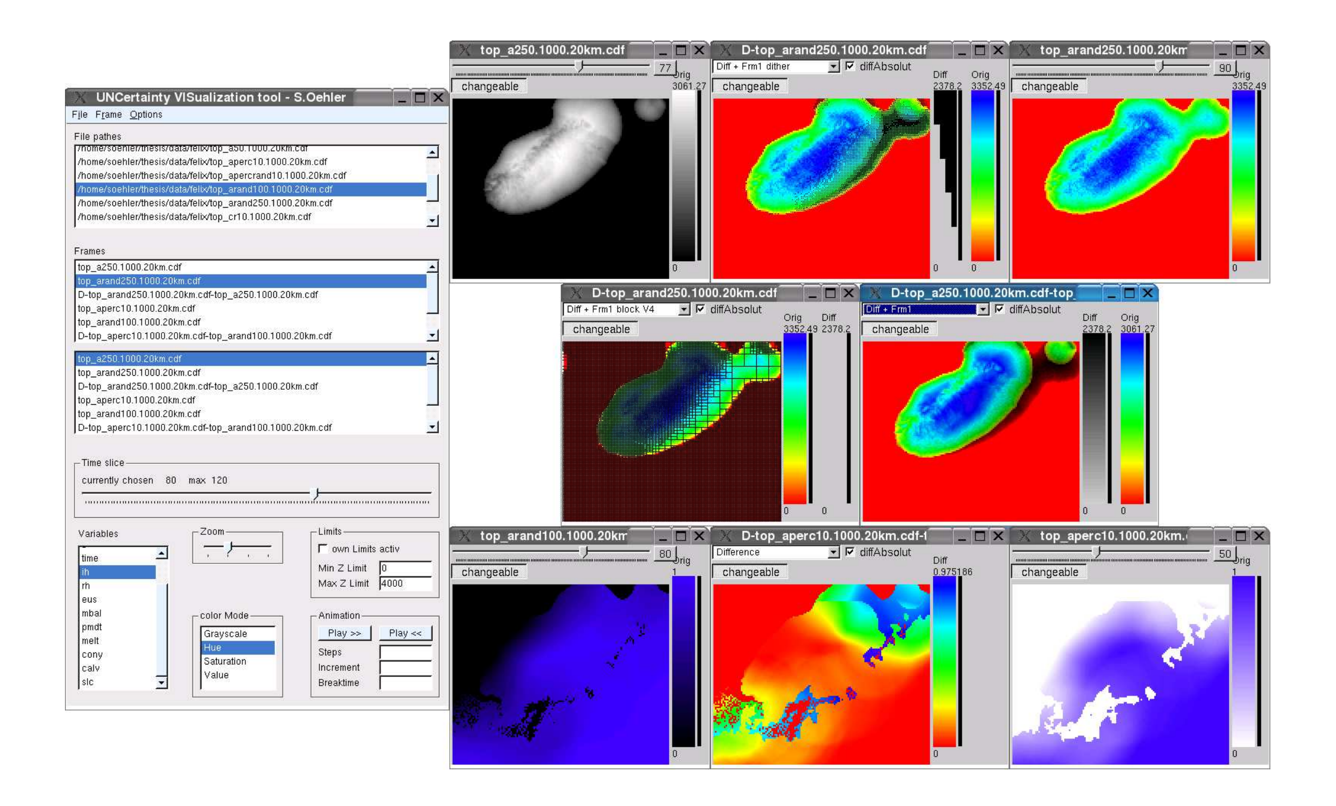
Visualisierung von Unsicherheiten unter Einbezug grossräumiger Modellierungsdaten



Sascha Oehler

DIPLOMARBEIT

Geographisches Institut der Universität Zürich

Zürich, September 2005

Betreuung: Dr. Ross S. Purves
Dipl. Felix Hebeler

Antragsstellendes Fakultätsmitglied: Prof. Dr. Robert Weibel

Vorwort

Zwei wichtige Bestandteile meines alltäglichen Erlebens konnte ich ihn dieser Diplomarbeit auf wissenschaftliche Weise verarbeiten. Zum Einen ist dies die wunderbare Fähigkeit des Sehens, die es dem Menschen ermöglicht, seine Umwelt mitttels den Augen zu erfahren und zu geniessen. Zum Anderen die Ungewissheit, Unvorhersehbarkeit und Unsicherheit, die in jedem Augenblick des persönlichen Erlebens der Umwelt präsent sind. Sie zeigen sich durch die Stärke der Veränderungen, der Um- und Neugestaltung der Gesamheit der geographischen Phänomene durch die Kräfte der Natur und der Menschen.

Ich bin überzeugt, das die Fähigkeit, sich auch im Alltag der unterschiedlichen Interpretationsmöglichkeiten der Situationen und Objekte in der Erlebenssphäre bewusst zu sein, eine gute Hilfe fürs Leben bieten kann. Es wirkt allfälliger Orientierungslosigkeit entgegen und vermag so das eigene Wohlbefinden zu stärken. Daraus lässt sich Ruhe und Kraft schöpfen. Dies bedingt allerdings, dass man die Chancen und Risiken jeder Option akzeptiert und unvoreingenommen analysiert, um diese gegeneinander abzuwägen.

Es ist befriedigend zu sehen, wie diese beiden Aspekte mit der Thematik der ‚Visualisierung von Unsicherheiten' sinnvoll vereint werden können. Gleichsam war es immer wieder notwendig, mich während des Schreibens der Arbeit auf die beiden Auslöser zurückzubesinnen.

Ohne die Unterstützung und Mithilfe zahlreicher Personen wäre diese Diplomarbeit nie entstanden. Sie waren stets zur Stelle, wenn ich wieder einmal am Sinn der Sache zweifelte oder ‚kein Licht am Ende des Tunnels' entdecken konnte. Ihnen allen sei gedankt.

Besonderer Dank geht an Dr. Ross Purves als meinen direkten Ansprechpartner für die intensive Betreuung und die stets sehr konstruktiven, zielgerichteten Hilfestellungen. Ein weiteres Dankeschön richtet sich an Felix Hebeler für das Bereitstellen der Modellierungsdaten und die konkreten Anregungen zum Design der Applikation. Ebenfalls danken möchte ich Prof. Dr. Robert Weibel für die gute Ausbildung und die unkomplizierte Endbetreuung dieser Diplomarbeit. Allen dreien herzlichen Dank für ihre Geduld, Flexibilität und die aufmunternden Worte zwischendurch.

Privat sind viele Menschen durch ihre Anteilnahme am Gelingen der Diplomarbeit mitbeteiligt. Speziell erwähnen möchte ich meine Eltern, die mir seit meiner Geburt eine optimale Rückendeckung bieten, die mich vorbehaltlos unterstützen und meine Entscheidungen und Verhaltensweisen herzlich akzeptieren. Last but not least ein ganz grosses Dankeschön an meine Freundin Simone Wyss für ihre Unterstützung in dieser für beide sehr schwierigen Lebensphase. Ich schätze und bewundere ihren Einsatz und ihre Geduld bei der Betreuung unserer Tochter Nora sehr.

Zusammenfassung

Fehlerfreie Modelle gibt es nicht. Limitierungen können beim Messen der Eingabeparameter oder dem Nicht-Kennen aller das Modell beeinflussender Parameter auftreten. Da die Modellresultate für Entscheidungsfindungen verwendet werden, ist es wichtig, die Entscheidungsträger mit den Unsicherheiten des betreffenden Modells zu beliefern.

Diese Diplomarbeit untersucht verschiedene Visualisierungsmöglichkeiten von Datenunsicherheiten. Mit Datenunsicherheiten sind Unterschiede oder Werteschwankungen in Resultatdaten gemeint, die beispielsweise aus Modellierungen stammen, die mit unterschiedlichen Eingabeparametern differierende Resultatesets liefern, wie es beim Einsatz der Monte Carlo Simulation üblich ist.

Datenunsicherheiten respektive -ungenauigkeiten oder -unschärfen entstehen im gesamten Ablauf der Datenprozessierung. Sie sind unvermeidbar mit jeglicher Art von Daten verknüpft. Eine Auswahl von Herkunftsquellen werden im Grundlagenkapitel präsentiert. Das angewendete theoretische Gerüst für die Unsicherheitsvisualisierung basiert auf den Bertinschen Grafikvariablen. Weiterführend werden neuere Grafikvariablen und komplexere Visualisierungstechniken analysiert. Es zeigt sich, dass eine grosse Auswahl an Visualisierungsoptionen denkbar sind, aus denen einige aufgegriffen werden.

Das Anliegen dieser Diplomarbeit ist, einen konkreten Lösungsvorschlag zu liefern, bestehend aus einer eigenständigen Applikation. Diese Applikation wird im Umsetzungsteil detailliert beschrieben. Sie beherrscht in der aktuellen Version die Darstellung originaler Datensätze und die Herstellung von Differenzen-darstellungen zweier solcher Datensätze. Als Beispieldaten kommen Eisschild-modellierungen des Fennoskandinavischen Eiskörpers zum Einsatz, die im NetCDF Format vorliegen. Aus diesem Grund ist die Datenschnittstelle der Applikation zur Zeit auf dieses Format beschränkt, die Erweiterung der erarbeiteten Konzepte auf andere datenformate stellt jedoch keine grossen Probleme dar.

Zur Überprüfung der erdachten Visualisierungen und der Applikation als solche, wurde eine Evaluation durchgeführt. Diese wurde im Rahmen eines offenen Interviews gestaltet, kombiniert mit einem kurzen, standardisierten Fragebogen. Zum Kennenlernen der Applikation erhielten die Tespersonen verschiedene Aufgaben, die sie mit Unterstützung des Testleiters zu lösen hatten. Die Resultate dieser Evaluation dürfen als sehr positiv eingestuft werden. Sowohl die Applikation als auch die Visualisierungen und deren Konzept werden als interessant und bereichernd bewertet. Die grösste Akzeptanz erzielen die einfachen bivariaten Visualisierungen, die in sekundenschnelle einen guten Überblick über die Daten und deren Unsicherheiten ermöglichen. Komplexere Visualisierungen sind weniger beliebt. Die Applikation selber kann durch die Implementierung weiterer Visualisierungs-optionen und Funktionalitäten verbessert werden. Besonders die GUI muss überarbeitet werden.

Summary

Error free models do not exist. There are always limitations in either mesuring the input parameters or by not knowing all parameters influencing the model. Since the result output of the models is used in decision making, it is very important to provide the decision makers with all known uncertainties concerning the specific model.

This diploma thesis examines different visualisation possibilities of data uncertainties. The term data uncertainties describes differences or value fluctuations in various output data sets. These differences originate, for example, from models which supply differing result sets from different input parameters, as it is usual with the employment of Monte Carlo Simulation.

Data uncertainties or data inaccuracies arise throughout the entire sequence of data processing. They are unavoidably linked to any kind of data. A selection of several sources of data uncertainty are presented in the chapter ‚Grundlagen' (fundamentals). The theoretical framework applied for the visualization of uncertainties is based on the famous graphic variables described by Jaques Bertin in 1967. Newer graphical variables and more complex visualization techniques are also analyzed. A large amount of visualization options are conceivable, some of which are tackled in the chapter ‚Implementierung' (implementation).

The central aim of this diploma thesis is to provide a concrete solution to the problem by implementing a bespoke application. This application is described in detail in the regarding chapter. In its actual version the application is able to produce representations of original data records and differences between two of such original data records. Data sets of ice sheet models of the fennoscandian ice body are used as sample data. They are available in the netCDF format, to which the data interface of the application is limited. An extension of the application to other data formats should not cause large problems.

For the evaluation of the implemented visualizations and the application itself, an evaluation exercise was undertaken. This was arranged in the context of an open interview, combined with a short, standardized questionnaire. For becoming acquainted with the application, the test persons received different tasks. They had to solve them with support of the test leader. The results of this evaluation can be classified as highly positive. The application, the visualizations and the visualisation concept are evaluated as interessting and useful. The highest acceptance could gain the most simple bivariate visualisation, which can give a good overview of the data and their uncertainties within a few seconds. More complex visualisations are less popular. The application itself can be improved by implementing more visualisation options and more functionality. Especially the GUI should be adjusted.



Inhaltsverzeichnis

Abbildungs- und Tabellenverzeichnis

Abbildungen

Tabellen

1 Einleitung

1.1 Motivation und Problemstellung

Es ist ein Bestreben des Menschen, die Geheimnisse der Welt um ihn herum zu entschlüsseln. Die möglichst originalgetreue Nachbildung mit Modellen ist dabei eine mögliche Vorgehensweise, sie zu ergründen. Es werden autonome Teilbereiche des Erdsystems separiert und diese mittels miteinander interagierender Parameter beschrieben. Diese Beschreibungen fliessen in Computersimulationen ein, die weitere Systemzustände berechnen. Modelle sollen helfen, zukünftige Situationen zu antizipieren. Ein Beispiel dafür sind unterschiedlich komplexe Klimasimulationen.

Ein Modell vermag die Realität nur bis zu einem gewissen Detaillierungsgrad zu repräsentieren, egal welche Art von Modell betrachtet wird. Es ist nicht möglich, alle Einflussfaktoren exakt zu reproduzieren, da einige nicht eindeutig bestimmbar sind. Somit müssen diese Modellparameter geschätzt werden. In manchen Fällen lässt man die Parameter ganz weg, bis sie bekannt und definierbar sind und nachträglich ins Modell eingefügt werden. Es entsteht so eine Annäherung an die echte Welt. Aus diesen Gründen sind Unschärfen und Zweideutigkeiten in den Modellen enthalten, die als Unsicherheiten zusammenfassbar sind. Unglücklicherweise sind Unsicherheiten allgegenwärtig und auch Geografischen Daten innewohnend (Foody 2003: 115 bezugnehmend auf Fisher 2000 und Bennet 2001). Diese können unter Umständen erheblich sein und ein Modell entscheidend beeinflussen. Deshalb sollten Angaben zu diesen Unterschieden mitgegeben werden. Die beste Form dafür ist, dem Resultateset eine umfassende Übersicht beizufügen. Für einen Rasterdatensatz wäre das beispielsweise ein deckungsgleiches Unsicherheitsraster, das für jede Rasterzelle den Schwankungsbereich angibt. Dies wird jedoch nur selten so detailliert gehandhabt. Über viele Jahre reichte es den Modellierern, den Karten ein kleines Zuverlässigkeitsdiagramm mitzugeben, um lokale Qualitätsschwankungen anzugeben. Eine anderes Verfahren ist eine schriftliche Zusammenfassung über die Modellunsicherheit als Ganzes (van der Wel et. al. 1994: 313). Oder gemittelte Resultate werden als das korrekte Abbild der Realität betrachtet. Dabei ist dies nur ein mögliches Szenario: es ist weder unumstösslich noch gilt es mit absoluter Sicherheit.

Um die Tragweite dieser Feststellung darzulegen, muss der Einsatzort einiger Modellierungsergebnisse mit einbezogen werden. Im Idealfall dient die Herstellung der Modelle keinem Selbstzweck, sondern sie sollen den Erkenntnisprozess fördern und Entscheidungsgrundlagen schaffen. Darauf stützen sich Entscheidungsträger, beispielsweise Politiker oder Wirtschftsführer, um ihre Entscheidungen zu fällen (Bertin 1983 und Foody 2003: 115). Die Modelle sollen helfen, Chancen und Risiken einer Situation besser abzuschätzen, um die bestmögliche Vorgehensweise zu beschliessen. Dieser Vorgang hat oft in kurzer Zeit zu erfolgen. Entscheidungsträger wollen Fakten, auf die sie sich stützen können. Und Unsicherheiten sind Fakten, bei manchen Diskussionsgegenständen sogar die entscheidenden. Je anschaulicher sie präsentiert werden, je schneller und einfacher sie erfassbar sind, um so mehr Beachtung wird ihnen geschenkt. Sie sollen dem Betrachter quasi ins Auge springen. Und Visualisierungen bieten das Potential, viel Information kompakt und zeitsparend aufzubereiten. Der Mensch visualisiert von Natur aus. Man braucht bloss sich selber als Beispiel zu nehmen, und es leuchtet ein, das eine Visualisierung viel einfacher zu verstehen und gebrauchen ist, als die selbe Information in einer tabellarischen Auflistung (Geldermans

& Hoogenboom 2001: 6). Ein bekanntes Sprichwort könnte leicht abgewandelt etwa lauten: Eine Modellvisualisierung sagt mehr als tausend Worte. Deshalb wäre es besser, wenn die Beschreibung der Unsicherheit nicht nur in schriftlicher Form vorliegt, wo sie eventuell gar nicht gelesen oder nur schnell überflogen wird, sondern ebenfalls als Grafik visualisiert aufbereitet ist. Dies bedingt allerdings eine vorgängige genaue Quantifizierung der Unsicherheiten.

Wie das Beispiel des Zuverlässigkeitsdiagrammes von van der Wel et al. (1994: 313) zeigt, ist das Wissen um die Problematik der Datenqualität den Geographen schon viele Jahrzehnte bekannt. Seit über drei Jahrzehnten, in denen GIS im Einsatz stehen, war Unsicherheit immer ein beachteter Gegenstand (Foody 2003: 115). Foody vermutet, dass die Aufmerksamkeit in den letzen Jahren leicht zunahm. Es werden immer wieder neue Konzepte zu deren Berechnung vorgestellt, und neue Definitionen des Unsicherheitsbegriffes eingeführt. Noch nicht ganz so lange wird eine Visualisierung der Unsicherheiten postuliert. Es finden sich einige Paper, die Visualisierungs-Methoden ansprechen. Trotzdem erhielt die Thematik nie den Stellenwert und die Aufmerksamkeit, die man ihr zugestehen könnte. Bereits vor 14 Jahren traffen sich einige spezialisierte Forscher in den USA, um sich dem Thema ‚Visualization of Data Quality' anzunehmen (Beard et al. 1991). Daraus entstammt eine interessante Ideensammlung. Leider konnten keine Forschungsarbeiten entdeckt werden, die darauf aufbauen. Nur ein paar wenige Autoren nehmen Bezug auf die Zusammenkunft.

Das Thema der Visualisierungen wurde in der Geographie mit dem zunehmenden Einsatz von Computern stetig populärer. Mit bescheidenen technischen Mitteln ist es heute möglich, Resultate in ansprechender Form zu präsentieren. MacEachren und Kraak (2001a), die beiden Herausgeber einer Spezialausgabe des Journals ‚Cartography and Geographic Information Science', stellen zusammen mit ihren Ko-Autoren eine ganze Forschungsagenda zu dieser Thematik auf. Die Forschergruppe um Fairbairn (2001), die sich vertieft mit Darstellungen und Kartographischen Visualisierungen auseinandersetzte, erwähnt in ihrem Bericht explizit die Wichtigkeit der Datenqualität respektive Datenunsicherheit und regt zur Forschung betreffend deren Visualisierung an. Sie sprechen dabei von einer essentiellen Aufgabe, Informationen über Datenunsicherheit und Metadatenparameter zur Verfügung zu stellen, um informierte Entscheidungen treffen zu können. Die Herstellung der dazu benötigten Visualisierungsinstrumente, die Erweiterung des Werkzeugkasten sei dabei eine entscheidende Arbeit. In allen 5 Artikeln des Journals werden die Leser ermutigt, sich den zahlreichen offenen Forschungsfragen bezüglich Geovisualisierungen anzunehmen.

Aus den genannten Gründen entstand die Idee, diese Diplomarbeit dem Thema Visualisierung von Unsicherheiten zu widmen. Die Faszination des Autors für visuelle Darstellungen per se, und die Visualisierungen von (Daten)Landschaften im Speziellen, fördern das Vorhaben dabei zusätzlich. Mit der Einarbeitung in die Materie stellte sich heraus, dass sie ein grosser Themenkomplex darstellt, der viele verschiedene Problematiken anspricht. Mit dem Visualisieren wird der Bereich der Kartographie betreten. Gesichtspunkte der Formensprache, der Farbkomposition, der Symbolik oder auch der Elementgrössen müssen berücksichtigt werden. Der Unsicherheitsteil wiederum verlangt mathematische und statistische Kenntnisse. Zuvor muss aber in einem fast philosophischen Sinn die Bedeutung des Begriffes geklärt werden. Es ist dem Autor ein wichtiges Anliegen, einen praktischen Teil in diese Arbeit zu integrieren. Eine weitere Leidenschaft, nebst dem Erfassen und Verfolgen optischer Eindrücke, ist der Informatiksektor als Ganzes und da vor allem die Softwareentwicklung. Der Aufbau der Diplomarbeit ist deshalb so gewählt, dass die Entwicklung einer eigenen Applikation eine sinnvolle Praxisergänzung zu den theoretischen Konzepten bietet.

1.2 Hypothesen und Fragestellungen

Der Mensch nimmt einen Grossteil seiner Wahrnehmungen und der daraus gewonnenen Informationen über die Augen auf. Sie sind seine primäre und direkteste Verbindung zur Aussenwelt. Ausgehend von dieser Überlegung wird folgende Hypothese aufgestellt:

„Die optische Sichtung von Resultatdaten unterschiedlicher Modellrechnungen ermöglicht einen raschen Überblick über deren Unterschiede respektive Unsicherheiten.“

Diese Hypothese dient als Leitmotiv für die Diplomarbeit und bildet ihren roten Faden, der stets als Hintergrund präsent bleibt. Als Sekundärziel soll diese Arbeitshypothese, soweit dies überhaupt möglich ist, validiert werden. Zur weiteren Spezifizierung dieser Arbeitshypothese und als Konkretisierung für die Umsetzung wird die folgende Anforderung angefügt:

„Bei der Visualisierung der Daten sollen die Stärke und die Verteilung der Unsicherheiten deutlich hervortreten und intuitiv verstanden werden.“

Aus der abstrakten und allgemein gehaltenen Leitidee sind 4 konkrete Fragestellungen abgeleitet.

I) „Wie kann man Unsicherheiten in Rasterdaten darstellen?“

Frage I zielt auf theoretische Überlegungen zu den postulierten Darstellungen. Bereits klingt dabei die praktische Umsetzung an, in der mit Rasterdaten gearbeitet wird. Dies aus dem Grund, dass für den konkreten Test diese Art von Daten – im Gegensatz zu Vektordaten – zur Verfügung stehen.

II) „Welche grafischen Variablen – beziehungsweise Visualisierungstechniken – sind für diese Darstellungen besonders geeignet?“

Frage II präzisiert die erste, indem von den verwendbaren Methoden gesprochen wird. Die beiden letzten Fragestellungen drehen sich um die praktische Umsetzung in der zu entwickelnden Applikation.

III) “Was sind die Anforderungen an eine Applikation dieses Typs?“

Frage III erkundet die generellen Bedürfnisse und Anforderungen an die zu entwickelnde Software. Dabei können zwei verschiedene Anforderungsarten unterschieden werden: Zum Einen die bereits vor der Implementierung abschätz- und planbaren. Zum Anderen jene Anforderungen die erst im Nachhinein als Erkenntnisse anfallen und den Lernprozess der Implementierung aufzeigen.

IV) „Wie urteilen Testpersonen über das Programm und die Darstellungen?“

Frage IV erweitert den Aktionsrahmen der Erkenntnissuche auf potentielle Nutzer und deren Perzeption der Applikation. Dabei steht die gesamtheitliche Betrachtung der Resultate im Vordergrund.

1.3 Aufbau und Ziele der Arbeit

Der Aufbau dieser Diplomarbeit gliedert sich in 6 Kapitel. Nach dieser Einleitung wird in Kapitel 2 auf den theoretischen Hintergrund eingegangen. Dabei wird erläutert, was mit Visualisierung gemeint ist und wo die Geovisualisierung im Speziellen steht. Ein weiterer Abschnitt ist dem Thema Unsicherheit gewidmet. Neben den unterschiedlichen Begriffsdefinitionen wird auf die Quellen von Unsicherheiten eingegangen. Als wichtige Bestandteile der Betrachtung von Visualisierungen werden die Grafikvariablen und verschiedene Visualisierungstechniken vorgestellt. Ihr Einsatz zur Darstellung von Unsicherheiten ist dabei zentral. Das kurze Kapitel 3 ist den verwendeten Beispieldaten gewidmet. Das eingesetzte Model wird beschrieben und die Unterschiede der einzelenen Datensets erwähnt. In Kapitel 4 erfolgt die Beschreibung der Umsetzung der theoretischen Überlegungen in eine Applikation. Nach der Beschreibung der zu Grunde liegenden Techniken wird der Applikationsaufbau, deren Benutzeroberfläche und Funktionalitäten betrachtet. Ein grosser Teilabschnitt befasst sich mit den implementierten Visualisierungen. Sie werden einzeln mit dem zugrunde liegenden Konzept und den benutzten Grafikvariablen vorgestellt. Die detaillierte Besprechung der Applikationsevaluation erfolgt in Kapitel 5. Darin wird auf den Testaufbau eingegangen und die Gruppe der Testpersonen vorgestellt. Ihre Antworten zum Standardfragebogen werden grafisch dargestellt. Mit jeder Testperson wurde zusätzlich eine freie Befragung durchgeführt, deren Ergebnisse ausführlich aufgelistet werden. In Kapitel 6 findet die Diskussion der Diplomarbeit statt. Dabei werden die Hypothese und die verschiedenen Fragestellungen aufgegriffen. Diese bilden den Bogen für die kritische Betrachtung der Arbeit. Dabei sollen das theoretische Konzept, die Applikation und die Evaluation beleuchtet werden. Im letzten Kapitel werden als Ausblick weiterführende Forschungsmöglichkeiten angedacht. Zukünftige Erweiterungen und Anpassungen der Applikation werden ebenso skizziert wie ein verbessertes Entwicklungsvorgehen und adäquatere Metadaten. Im Anhang schliesslich finden sich das Quellenverzeichnis, der in der Evaluation benutzte Fragebogen, eine CD mit den für die Installation der Applikation benötigten Dateien und ein Auszug aller Header-Datei des Applikations-Quellcode.

Dieser schriftliche Bericht hat zum Ziel, die theoretischen und praktischen Erkenntnisse, die durch das Literaturstudium und die Implementierung der Software gewonnen wurden, in einer optisch ansprechenden Form zu gruppieren und präsentieren. Der Leser soll einen Einblick in die Materie der Visualisierung von Unsicherheiten erhalten und die Grundzüge der Thematik verstehen. Dabei werden einige ausgewählte Aspekte stärker beleuchtet als andere. Die Applikation ist ein Beispiel für eine mögliche Lösung der Problemstellung. Zusammen mit der Evaluation bildet sie für Interessierte eine Basis, auf der sie eigene Implementierungen aufbauen können. Dabei dürfen die auf der CD vorliegenden Klassen frei verwendet und modifiziert werden. Dies unter der Bedingung, dass die Weiterentwicklungen ebenfalls der Öffentlichkeit frei zur Verfügung gestellt werden. Der Autor freut sich auf Anregungen und Anfragen zu allen Aspekten dieser Diplomarbeit und stellt sich für nähere Auskünfte gerne zur Verfügung.

2 Grundlagen

2.1 Einführende Begriffsdefinitionen

GIS

GIS ist die Abkürzung für Geographische Informationssysteme. Die vorliegende Diplomarbeit ist eng mit dem Gebiet der GISysteme verbunden. Bei der weiteren Beschreibung und Definition des Untersuchungsgegenstandes wird auf sie Bezug genommen. Deshalb folgt eine Definition und Beschreibung derselben. Eine werkzeugbezogene Definition beschreibt ein GIS als ein System zur Aufnahme, Speicherung, Überprüfung, Manipulation, Analyse und Anzeige von georeferenzierten räumlichen Daten (Departement of Environment 1986, in Burrough and McDonnell 1998). Eine datenbankbezogene Definition beschreibt ein GIS als Datenbanksystem, in dem der grösste Teil der Daten räumlich indexiert sind. Auf ihr operieren ein Set von Prozeduren zur Beantwortung von Anfragen über räumliche Entitäten aus der Datenbank (Smith et al. 1987, in Burrough and McDonnell 1998). GIS sind also dazu da, räumliche Daten zu speichern, strukturieren und Hilfsmittel zur Manipulation und Darstellung dieser Daten zur Verfügung zu stellen. Diese Diplomarbeit und die dazu implementierte Applikation verfolgen die gleiche Absicht: Es soll ein Datenzugriff auf räumliche Daten erfolgen, um diese auf eine spezielle Art darzustellen.

Raster(daten)

Diese Diplomarbeit untersucht hauptsächlich Rasterdaten. Ebenso kann die implementierte Applikation nur Rasterdaten verarbeiten und darstellen. Rasterbasierte räumliche Modelle betrachten den Raum als Tessellation von Zellen, denen allen ein Eintrag einer Klassierung oder eine Identität eines Phänomens, das die Zelle besetzt, zugeordnet ist. Der Ausdruck Pixel (Abkürzung von picture element) wird dabei oft einer Rasterzelle zugeordnet. Beide Begriffe stammen aus dem Bereich der Bildverarbeitung, wo die einzelnen Bilder mit einem Rasterscanning-Prozess entstehen, der mit Videobildern oder Fernsehkameras verglichen werden kann (Jones 1997: 33). Raster repräsentieren die zweidimensionalen Platzierungen von Phänomenen als Matrix von Gitterzellen (Jones 1997: 35). Rasterzellen sind typischerweise quadratisch.

Das objektgerichtete Gegenstück zu den Rastermodellen sind Vektormodelle. Dabei werden alle Phänomene als primitive oder zusammengesetzte räumliche Entitäten aufgefasst (Jones 1997: 30).

Visualisierung

Eine kurze und zugleich sehr treffende Begriffsdefinition von Visualisierung liefern Fuhrmann und Kraak (2001: 173): „Visualisierung bedeutet etwas sichtbar machen“. Diese Diplomarbeit nutzt Visualisierungen um modellierte Daten über untersuchte Phänomene geordnet darzustellen. Die Visualisierungen werden zur Reduktion der Komplexität von dargestellten Informationen benutzt (Geldermans und Hoogenboom 2001). Unterschiedliche Medien sind dazu als Informationsträger verwendbar. Sehr alte Darstellungen sind in Höhlen auf Felsen gemalt erhalten. Die Menschen setzten weitere Materialien wie behauene Steine, Tierhäute und -knochen, Papyrusrollen, Metallplatten und Papier ein. Seit der Entwicklung und weiten Verbreitung von Computersystemen werden die Visualisierungen zu einem grossen Teil von spezialisierten Softwarepaketen generiert und an

Bildschirme projiziert. Dabei muss der Benutzer dem System über Bedienelemente wie Tastatur und Maus vorgeben, wie es die Informationen verarbeiten soll. Dieser Vorgang bietet den Vorteil, dass vor dem Übertragen auf ein anderes Betrachtungsmedium (meistens Papier) Varianten der Informationsdarstellung erprobt werden können.

Der allgemeine Begriff Visualisierung leitet zum spezifischeren Begriff der Geovisualisierung im nächsten Kapitel über.

2.2 Geovisualisierung

Obschon der Begriff Visualisierung im Titel der Arbeit verwendet wird, ist diese spezifischer im Gebiet der Geovisualisierung anzusiedeln. Dieser Begriff steht für eine Untermenge der Visualisierung, setzt er sich doch aus den zwei Teilen Geographie und Visualisierung zusammen und bezeichnet demnach Visualisierungen unter einem spezifisch geographischen Blickwinkel. Crampton (2002: 85) definiert den Begriff so, dass Geovisualisierung eine Methode und ein Ansatz zur Visualisierung von geographischen Daten ist. Dabei können Muster erforscht, Hypothesen generiert, Verbindungen und Lücken in Daten erkannt und Trends identifiziert werden. Er bemerkt ausserdem, dass speziell darauf hingewiesen werden muss, dass GVis (wie er Geovisualisierung abkürzt) nicht einfach die richtigen Lösungen präsentiert, sondern ein Set von Werkzeugen bereitstellt, um Daten auf verschiedenen Wegen zu beleuchten. Bei seiner Definition stimmt er mit Kraak (2000: 27) überein, der schrieb: ‚Geovisualisierung kann als Nutzungsfeld von visuellen geografisch-räumlichen Darstellungen einschliesslich virtueller Umgebungen definiert werden, um Daten zu erforschen und dadurch Fragen zu beantworten, Hypothesen aufzustellen, Problemlösungen zu entwickeln und Wissen zu generieren.’ Beide Forscher stellen einen klaren Bezug zum Nutzen dieser Darstellungen her. Sie beschreiben die Geovisualisierung als eine Unterstützung bei der Wissensgenerierung, als Instrument zur Aufbereitung von Daten und deren vereinfachter Sichtung. Kraak beschreibt weiter, dass Karten in ihren vielen unterschiedlichen Erscheinungsformen dabei eine Schlüsselrolle spielen. Dazu findet sich bei Fuhrmann und Kraak (2001: 173) folgende Präzisierung: „Karten und kartographische Abbildungen werden nicht mehr für reine Präsentationszwecke genutzt, sondern auch zur Exploration von temporalen und nicht-temporalen Geodaten eingesetzt.“ Dass dabei weniger die klassische Form von Karten als gedruckte Produkte gemeint sind, sondern moderne, computergestützte GIS, zeigt sich in einer anderen Aussage, die MacEachren zusammen mit Kraak (2001: 3) in einem Paper äussert: ‚Karten und Grafiken vermögen in diesem Kontext mehr zu leisten als blosses Sichtbarmachen der Dinge. Sie sind aktive Instrumente im Denkprozess der Nutzer.’ Der Begriff Geovisualisierung bedeutet demnach für sie, dass Ansätze von Visualisierungen in Computerwissenschaften, Kartographie, Bildanalyse, Informationsvisualisierung, Explorativer Datenanalyse und Geografischen Informationssystemen integriert werden, um Theorie, Methoden und Werkzeuge zur visuellen Exploration, Analyse, Synthese und Präsentation von geografischen, räumlichen Daten bereitzustellen. Kürzer und prägnanter ist eine Aussage etwas später im selben Text: Heutige kartographische Umgebungen sind durch zwei Schlüsselwörter charakterisierbar: Interaktion und Dynamik (als Gegensatz zu statischen Karten verstanden). Kraak (2000: 27) weist darauf hin, das mit diesem neuen Verständnis der geografischen Datenaufbereitung mittels Visualisierung auch eine andere Sichtweise auf das Kartendesigns eingenommen wird.

2.3 Unsicherheit

Mit diesem Begriff wird ein weites und heterogenes Bedeutungsfeld belegt. Man kann sich unsicher fühlen, weil etwas Unbekanntes, Unerwartetes eintritt. In diesem Beispiel steht der Begriff für die eher negativ beladene Emotion des ‚sich in der Luft' fühlen. Oder wir sagen, das Wetter sei unsicher, es könnte Regen geben. Hierbei drückt sich ein Zweifeln aus, eine Einschätzung, der wir nicht vertrauen. Es gibt auch den Ausdruck der unsicheren Wohnlage, weil sie durch Umwelteinflüsse bedroht wird, oder die Kriminalität dort statistisch besonders hoch ist. Diesen drei Beispielen gemein ist eine mehr oder minder grosse subjektive Komponente. Erst der Akteur empfindet eine Situation als unsicher. Eine andere Person mag eine andere Meinung dazu haben. Im Gegensatz zur Alltagsunsicherheit verwendet diese Diplomarbeit den Begriff für eine definierbare Unsicherheitskomponente in Daten beziehungsweise den datenliefernden Modellen. Es lassen sich gewisse Analogien ziehen. In allen Fällen ist das nicht exakt vorhersehbare Verhalten eines Systems Schuld an der Unsicherheit. Im Alltag ist dieser Umstand vielleicht unangenehm, in der Wissenschaft jedoch sollte das gar nicht vorkommen. Es ist in vielen Forschungsbereichen nicht zu vermeiden, mit Unbekannten zu arbeiten. Deshalb ist eine Quantifizierung derselben wünschenswert. Ist die Grösse der Unbekannten abschätzbar oder bekannt, kann die Toleranz eines Modells berechnet und angegeben werden. Dies bringt die Unsicherheit zwar nicht zum Verschwinden, macht sie aber kalkulierbar. Gahegan et al. (2001: 39) beschreiben dieses Phänomen wie folgt: Die zu visualisierenden Daten sind selten oder nie frei von Interpretationen und konzeptuellen Modellen, die ebenfalls das schlussfolgern beeinflussen. Sie stellen weiter die Frage, wie diese Modellunsicherheiten am adäquatesten den Nutzern präsentiert werden können.

2.3.1 Definition des Begriffes Unsicherheit

Über die Definition von Visualisierung respektive Geovisualisierung ist sich die Forschergemeinde ziemlich einig. Beim Unsicherheitsbegriff ist eher das Gegenteil der Fall. In der Literatur erscheint eine Vielzahl von Begriffen sowie die dazugehörenden Interpretationen. Nebst Unsicherheit gebrauchen die Forschenden auch Terme wie Mehrdeutigkeit, Unschärfe, Verschwommenheit, Abweichung, Fehler und Präzision, um nur eine Auswahl zu nennen. Für die vorliegende Arbeit stellt sich weiter das Problem, dass die gesamte Begrifflichkeit in englischer Sprache verfasst ist und nicht für alle Ausdrücke eine eindeutige deutsche Ein-Wort-Übersetzung möglich ist. Trotzdem wird versucht, deutsche Ausdrücke zu verwenden, solange sich die Bedeutung der englischen Originalbegriffe nicht stark verzerrt oder gar in ihrem Sinn umkehrt.

Atkinson und Foody (2002: 2) führen zu Beginn ihres Buches über Unsicherheit in der Fernerkundung ein kleines Vokabularium ein. Sie schlagen vor, Unsicherheit (uncertainty) als Oberbegriff für die Gesamtheit dieses Ungenauigkeitsphänomens zu verwenden, der sich in Mehrdeutigkeit (ambiguity) und Unklarheit/Verschwommenheit (vagueness) unterteilen liesse. Sie möchten den ersten Term für ‚Crisp Sets' verwenden, den Zweiten für ‚Fuzzy Sets' und ‚Rough Sets'. Zheng (2001: 310) beschreibt die Crisp Sets (scharfe Mengen) als Vertreter von Datenmengen der traditionellen Booleschen Logik, bei der Elemente entweder einer Klasse zugehören oder nicht. Zadeh (1965) führte die Idee der Fuzzy Sets (unscharfe Mengen) ein. Bei ihnen wird die Klassenzugehörigkeit nicht mit ja oder nein respektive 0 oder 1 angegeben, sondern mit einem bestimmten Grad der Sicherheit. Je nachdem bei welchem Schwellenwert die Klassengrenzen gezogen werden, gehören die untersuchten Element einer Klasse an oder nicht. Ahlqvist et al. (2000) beschreiben zusätzlich die Rough Sets, in denen obere und

untere Grenzen (Intervalle) für die Klassenzugehörigkeiten definiert werden. Befinden sich die Werte innerhalb dieser Intervalle, lassen sie sich der einen oder der anderen Klasse zuordnen. In einem weiteren Schritt kombinieren die selben Autoren diese Sets zu ‚Rough Fuzzy Sets' (Ahlqvist et al. 2003). Für die später beschriebenen Visualisierungen spielt es keine Rolle, aus welcher Mengentheorie die Daten genau entstammen. Ihre Datenstruktur sollte jedoch eine Abweichungsgrösse enthalten, die für die Darstellungen verwendet werden kann. Es gibt die Möglichkeit, eine Unsicherheit oder Wahrscheinlichkeit darstellerisch zu simulieren, indem zwei fixe Modelle (Crisp Sets) miteinander verglichen werden. Die Erzeugung der Modellunterschiede geschieht dabei z.B. mittels Monte Carlo Simulation. Zhang und Goodchild (2002: 5) argumentieren, dass der Begriff Unsicherheit den tatsächlichen Sachverhalt besser beschreibt als der Begriff Fehler (error). Letzterer impliziert, dass eine absolute Wahrheit existiert, die exakt bestimmbar ist, wenn die richtigen Messinstrumente zur Verfügung stehen. Das ist aber ein Trugschluss, da oftmals aus vielerlei Gründen bloss eine Annäherung an die Wirklichkeit gelingen kann. Aus dem selben Grund lehnen die Autoren auch die Begriffe Mehrdeutigkeit (ambiguity), Unexaktheit (inexactness) bzw. Verschwommenheit (vagueness) ab, die in vielen geographischen Studien benutzt werden. Auch diese suggerieren, dass es sich um eine blosse technische Ungenauigkeit, um einen (Mess)fehler handelt, der mit den richtigen Instrumenten vermeidbar ist. Da es sich jedoch um eine Informations- und Wissenslücke dreht, also um einen (momentan) nicht änderbaren Mangel innerhalb des verwendeten Modells, drückt Unsicherheit diesen Sachverhalt am besten aus.

Unsicherheit korreliert direkt mit der Datenqualität, die sich aus verschiedenen Arten von Datengenauigkeit zusammensetzt. Die DCDSTF 1988 (in Buttenfield 1991) differenziert fünf solcher Genauigkeitsdimensionen, die allgemein anerkannt sind: Die Abstammung (lineage), die Positionsgenauigkeit (positional accuracy), die Attributgenauigkeit (attribue accuracy) und die Vollständigkeit (completeness) können in einem Datenset lokal differieren, während die logische Konsistenz (logical consistency) nur für das Gesamtprodukt angegeben werden kann (van der Wel et al. 1994: 319).

Für die vorliegende Diplomarbeit ist die exakte Definition des Begriffes Unsicherheit zweitrangig. Wichtiger ist die damit verbundene Sichtweise auf die Daten und ihre Ausprägung. Es soll zum Ausdruck kommen, dass der Wertebereich der angesprochenen Modellresultate einen gewissen Spielraum beinhaltet. Um die angestrebten Visualisierungen umsetzen zu können muss dieser Spielraum definiert, berechnet und als Metadaten mitgeliefert werden.

2.3.2 Quellen von Datenunsicherheiten

Die Quellen von Datenunsicherheiten sind mannigfaltig und können in jedem Schritt des Forschungsprozesses auftreten. Je früher sich Ungenauigkeiten einschleichen, umso grösser ist normalerweise ihr Einfluss auf das Endresultat. Jeder Datenmanipulationsschritt ist eine potentielle Quelle, von der geografischen Abstraktion über die Akquisition und der Geo-Prozessierung bis hin zur Nutzung der Daten. Während des gesamten Datenverarbeitungsprozess findet eine Datenreduktion statt. Dies ist sogar nötig, um die Datenverarbeitung durchzuführen. Dabei werden den Daten allerdings Ungenauigkeiten und Unsicherheiten hinzugefügt (Zhang und Goodchild 2002). In der folgenden Auflistung werden einige Unsicherheitsquellen vorgestellt.

Geographische Daten unterliegen der **Abstraktion**, um ihre schiere Menge nutzbar zu machen. Dabei werden sie **vereinfacht und diskretisiert**, um eine Vereinfachung der Datenanalyse zu ermöglichen. **Konzeptionsfehler** bilden eine weitere Kategorie. Unsicherheiten entstehen **beim Einordnen von**

Elementen in Klassen, wenn die **Klassengrenzen unadäquat** gezogen werden oder ein Element eine **Übergangsstufe** zwischen den Klassen darstellt (Zhang und Goodchild 2002). Die **Wahl des Analyseverfahrens** schränkt die Datenaufnahme ebenso ein wie **Vereinfachungen und Annäherungen** bei den Situationsanalysen und den Definitionen der Untersuchungsobjekte (Foody und Atkinson 2001).

Oft sind die Messungen und das Verständnis der Daten unsicher. **Messunsicherheiten** treten häufiger auf als **Verständnisunsicherheiten**, haben aber eine kleinere Auswirkung (Foody und Atkinson 2001). **Messfehler** sind ebenfalls häufig. Sie können sowohl die **Position** von Objekten wie auch deren **Attribute** betreffen. Diese Fehler können durch den **unsachgemässen Einsatz von Messinstrumenten** entstehen oder durch deren **limitierte Messgenauigkeit** (Zhang und Goodchild 2002). Manche Messungenauigkeiten sind schwer zu vermeiden, da sie durch äussere Einflüsse bestimmt sind. Darunter fallen **seismischen Bodenbewegungen, Effekte der Kontinentaldrift** oder die **Lageänderung der Erdachse** (Longley et al. 2001).

Einige **technische Probleme** des Digitalisierens gedruckter Karten sind **Hardwarefehler**, **Bedienungsfehler** der Apparaturen durch den Operator, **endliche Liniendicke** auf den Originalkarten und **numerische Instabilitäten** in den Berechnungen (Dutton 1991). Weitere Beispiele maschinenbezogener Fehler sind **unpräzise Linienverfolgung** oder Differenzen bei Photogrammetrieplottern. **Menschliches Versagen** tritt beispielsweisse bei **Falschklassifikationen** auf. Gründe dafür sind **falsches Betrachten** oder **Interpretieren** der Untersuchungsgegenstände (Zhang und Goodchild 2002).

Das gedankenlose Zusammentragen möglichst grosser Datenmengen ergibt nicht zwingend genauere Ergebnisse. Grössere Datenbestände sind oft aus **unterschiedlichen Quellen** mit **unterschiedlichen Genauigkeiten, Detailausprägungen und Qualitäten** zusammengesetzt. Ältere Datenbestände können mit **unpräziseren Instrumenten** erfasst worden sein. Klassierungen können über viele Jahre konstant bleiben, oft geschieht allerdings das Gegenteil des **Neueinteilens und Redefinierens von Klassen**. Die verschiedenartige Herkunft wird beim Zusammenführen unter der Annahme von skalenlosen und hochpräzisen Daten missachtet (Zhang und Goodchild 2002, Longley et al. 2001). **Quellenunstimmigkeiten** und **unvollständige Informationen** sind auch für Foody und Atkinson (2001) wichtige Fehlerquellen.

Prozessierungsfehler sind unter anderem bei der **Umwandlung von Rasterdaten in Vektordaten und umgekehrt** zu orten. Die **Generalisierung** und **Skalierung** von Daten ist ein wichtiger Punkt bei der Suche und Vermeidung von Unsicherheiten. Mit den bereits in der Qualität reduzierten Daten werden schliesslich **vereinfachte Entscheidungsfindungen und Entscheidungsbeschlüsse** durchgeführt (Zhang und Goodchild 2002). Die **Art der Repräsentierung** der verarbeiteten Daten kann ebenfalls einen weiteren Qualitätsverlust bedeuten. **Sprachliche Ungenauigkeiten** im Sinne eines **unsachgemässen Sprachgebrauches**, der den Bedeutungsinhalt einer Information nicht eindeutig identifiziert oder diesen Inhalt gar verkehrt, sind in jedem Datenverarbeitungsschritt möglich (Foody und Atkinson 2001).

2.3.3 Unsicherheitsmodellierung mittels Monte Carlo Simulation

Die Monte Carlo Simulation (MCS) ist eine beliebte Methode, Unsicherheitsdaten zu erzeugen (Ehlschlaeger 2002: 259, Foody 2003: 115). Die Simulation trägt ihren Name durch eine Gemeinsamkeit mit der gleichnamigen Hauptstadt Monacos, Monte Carlo. Diese ist für ihre

Spielcasinos berühmt, in denen an Glücksspielen um Geld gewettet wird. Bei diesen Spielen entscheidet deren zufälliges Verhalten über Gewinn oder Verlust. Die Zufallskomponente ist auch der Kern der MCS (CSEP 1995). Bei einer MCS wird das gewählte Modell mehrmals mit unterschiedlichen Eingabegrössen durchgerechnet. Dabei darf eine Mindestanzahl Modellläufe nicht unterschritten werden. Die Eingabewerte sind gemäss einer Verteilungsfunktion, die zu bestimen ist, zufällig in einem Intervall gestreut. In diesem Fall wird von perturbierenden Eingabegrössen gesprochen. Auch die Modellresultate liegen schliesslich nach deren Berechnung als Zufallsvariablen vor. Die Gesamtheit der Modellresultate kann mit statistischen Methoden analysiert werden (Schöning 1996: 102). Daraus lässt sich schliesslich die Unsicherheit ableiten, beispielsweise unter Berücksichtigung der Streuung der Modellresultate.

Ehlschlaeger weist darauf hin, dass für die resultierenden Daten unterschiedliche Bezeichnungen angewendet werden, zum Beispiel Fehlermodell, Unsicherheitsmodell räumlicher Daten, Fehlerfortpflanzungsmodell oder auch Fehlertaste (error button). Letztgenannter Ausdruck demonstriert den Wunsch vieler Forscher, die benutze Software möge einen Knopf vorweisen, der bei Betätigung die Unsicherheit der bearbeiteten Daten präsentiert. Die Anwendungsunsicherheit kann durch sich unterscheidende Höhenmodelle simuliert werden (Ehlschlaeger 2002: 260). Eine zufällig variierende Geländeoberfläche hat auf einige Modelle einen bedeutenden Effekt. Genauso verhält es sich auch bei den Beispieldatensätzen für den Applikationstest dieser Diplomarbeit, die aus ebensolchen Oberflächenvariationen stammen und sich durch diese Modulation teilweise erheblich voneinander unterscheiden.

2.4 Visualisierung von Unsicherheitsdaten

2.4.1 Grafische Grundelemente

Sämtliche grafischen Darstellungen lassen sich in die gleichen elementaren visuellen Komponenten zerlegen. Diese erlauben die exakte Analyse des optischen Aufbaus der untersuchten Abbildungen. Zur besseren Vergleichbarkeit untereinander und zur gemeinsamen Nutzung eines einheitlichen Sprachgebrauchs wird ein standardisiertes Regelwerk zur Beschreibung dieser Komponenten angestrebt. Eine Mehrheit der Autoren, die sich mit Geovisualisierungen oder modernen kartographischen Darstellungen befassen, verweisen in ihren Untersuchungen auf den Pionier dieser Forschungsrichtung, Jacques Bertin. Dieser Forscher lieferte 1967 das Grundvokabularium zur Beschreibung grafischer Darstellungen. Er definierte sieben grafische Variablen, in die er alle Abbildungen zerlegt. Es sind dies die Grösse (size), die Form (shape), der Farbwert (value), die Orientierung, die Textur, der Farbton (color hue) und die Platzierung (location oder position) (Buttenfield 1991, Jones 1997: 253). Dieses Basisregelwerk wird heute noch angewendet und hat wenig von seiner Bedeutung eingebüsst. Die sieben Komponenten werden heute zu Bertins Ehren Bertin'sche Grafikvariablen genannt, um ihren Einfluss auf den ganzen Forschungsbereich angemessen zu honorieren (Blok 2000 und van der Wel et al. 1994: 320). Der Begriff Grafikvariable setzte sich allgemein als offizielle Bezeichnung für die grafischen Bausteine von Abbildungen durch. Einige Autoren definieren Bertin folgend ihrerseits neue Grafikvariablen um damit die klassischen Sieben zu ergänzen beziehungsweise zu verfeinern. Manche davon sind eine Umbenennung bereits beschriebener Grafikvariablen. Andere wiederum entstehen durch eine neue Betrachtungsweise der Bildkompositionen und bilden nicht nur eine Ergänzung, sondern eine Alternative zur Bertin'schen

Ordnung. Ein treffendes Zitat, das die hier betriebene Fokussierung auf die Grafikvariablen rechtfertig, ist in van der Wel et al. (1994: 319) zu finden. Sie schreiben: ‚Das Problem der Visualisierung von Datenqualität lässt sich auf folgendes Problem konzentrieren: Welche Grafikvariable kann in welchem Stadium des Prozesses der Informationsextraktion angewendet werden, um einen spezifischen Qualitätsparameter zu repräsentieren, der auf ein bestimmtes Datenset verweist, um den angepeilten Nutzen eines Anwenders aufzuzeigen.'

2.4.2 Die sieben Bertin'schen Grafikvariablen

Die sieben Bertin'schen Grafikvariablen werden nun einzeln vorgestellt und auf ihre mögliche Eignung zur Visualisierung von Daten mit ihren Unsicherheiten untersucht. Die betrachteten Untersuchungsgegenstände sind dabei hauptsächlich bivariate Darstellungen, in denen die Daten und die Metadaten über die Qualität gleichzeitig in einer Visualisierung gezeichnet werden. Eine tabellarische Auflistung dazu präsentieren van der Wel et al. (1994: 321). Darin klassieren sie 9 Grafikvariablen in die 3 Klassen ‚anwendbar', ‚anwendbar unter bestimmten Einschränkungen' und ‚nicht anwendbar'. Diese Unterteilung wird auf die vier Skalenniveaus nominal, ordinal, interval und rational angewendet und dabei jeweils separat nach den vier vom DCDSTF (1988) aufgestellten Datenqualitätsparametern Positionsgenauigkeit, Attributgenauigkeit, Abstammung und Vollständigkeit untersucht. Durch die Informationsfülle ist die Tabelle ziemlich unübersichtlich und schlecht lesbar. Die Autoren beschreiben ihr Konzept bei der erfolgten Einteilung, erläutern aber nicht die einzelnen Grafikvariablen. Dadurch kann die vorgenommene Einteilung teilweise nicht nachvollzogen werden. Sie unterlassen es, konkrete Anwendungsbeispiele vorzuschlagen oder zu skizzieren. Dies wird hier unter dem speziellen Fokus der im Umsetzungskapitel verwendeten Rasterdaten einer Eisschildmodellierung versucht. Dabei werden die Daten idealisiert betrachtet, ohne die Unterscheidung der Qualitätsparameter zu berücksichtigen. Das Ziel ist die Suche nach der rein visuell möglichen Anwendbarkeit der Grafikvariablen. Vektordaten stehen dabei im Hintergrund, werden aber nicht ausdrücklich ausgeschlossen.

Color hue (Farbton)

Mit dieser Grafikvariable wird die umgangssprachlich einfach Farbe genannte Grafikkomponente bezeichnet. Das Farbtonspektrum ist im Wellenlängenbereich des sichtbaren Lichtes angesiedelt und reicht vom kurzwelligen Violett bis zum langwelligen Rot. Im Gegensatz zu neueren Autoren unterscheidet Bertin nicht zwischen Farbsättigung (saturation) und dem Farbton. Diese Arbeit übernimmt aber die 1974 durch Morrison eingeführte und später anerkannte Trennung beider Grafikkomponenten (Blok 2000). Die Beschreibung der Farbsättigung erfolgt im anschliessenden Unterkapitel ‚Erweiterte Grafikvariablen'.

In kombinierten Darstellungen ist es möglich, die variierenden Differenzen mittels Farbtonmodulation darzustellen. Dies bedingt, dass die eigentlichen Daten mit einer der anderen Grafikvariablen variiert werden müssen. Am besten eignen sich dafür die Helligkeit (Farbwert / Farbsättigung) oder die Textur. Die Interpretation der erzeugten Darstellungen ist schwierig und kann zu Fehlschlüssen führen, weshalb sie zurückhaltend eingesetzt und sorgfältig hinterfragt werden sollten. Der Autor rät vom häufigen Gebrauch dieser Variablenkombinationen ab, da bessere und eindeutigere Varianten existieren (Abbildung 2-1).

Mit dem Farbton kann der alleinige Unsicherheitsgrad in Datensätzen sehr gut dargestellt werden. Allerdings entsteht dabei der Eindruck einer Klassierung, da der Farbton keine natürliche optische Hierarchie auszudrücken vermag. Deshalb ist der Einsatz bei Intervall- oder Rationaldaten eingeschränkt.

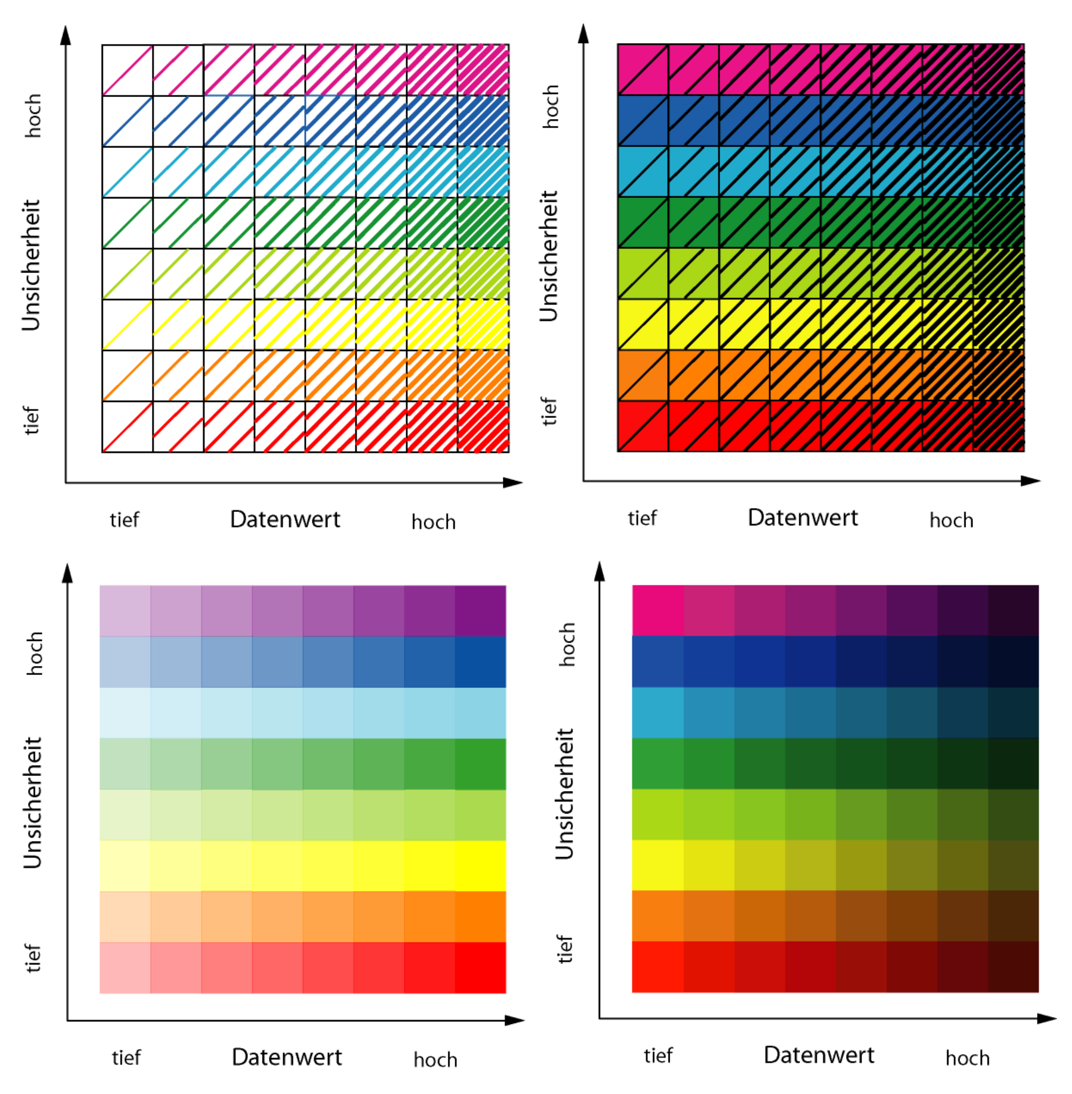


Abbildung 2-1 Unsicherheit mittels Farbton, Datenwerte mittels Textur (oben links und rechts), Farbsättigung (unten links) und Farbwert (unten rechts) visualisiert

Color value (Farbwert)

Der Farbwert ist geradezu prädestiniert für die Unsicherheitsdarstellung. Er definiert die Intensität (Helligkeit) des Farbtons. Objekte, deren Farbwert 0 ist, werden ganz in schwarz gezeichnet, bei einem Farbwert von 1 dagegen mit dem ausgesuchten Farbton. Damit lässt sich ein Abblendeffekt erzielen, der die Unsicherheit grafisch umsetzt: Je grösser die Unsicherheit der Datenwerte, desto dunkler werden diese Werte gezeichnet. So erscheinen unsichere Werte farblich vermindert, sind weniger gut erkennbar und feine Nuancierungen verschwinden in Übereinstimmung mit der weniger gesicherten Datenlage. In Kombination mit dem Farbton, der für die eigentlichen Datenwerte eingesetzt wird, ergeben sich schöne, klare Abstufungen. Zusammen mit einer Graustufendarstellung ist der Farbwert nicht einsetzbar. Grau- und Farbwert wirken direkt auf die Helligkeit der Bildelemente ein

und beeinflussen sich gegenseitig. Die Anwendung der Textureigenschaft mit dem Farbwert ist denkbar, aber problematisch. Je nach Intensität der Textur kann die Wirkung des Farbwertes verloren gehen. Dies ist bei besonders feinen Texturen der Fall (Abbildung 2-2).

Diese Grafikvariable kann für die alleinige Darstellung der Unsicherheitsdaten verwendet werden, indem man z.B. für höhere Werte tiefere (dunklere) Farbwerte einsetzt. Für den Einsatz in einer Einzeldarstellung ist die Auswahl eines zu variierenden Farbtons notwendig. Wird als Hauptfarbe weiss gewählt, erhält man eine einfache Graustufendarstellung.

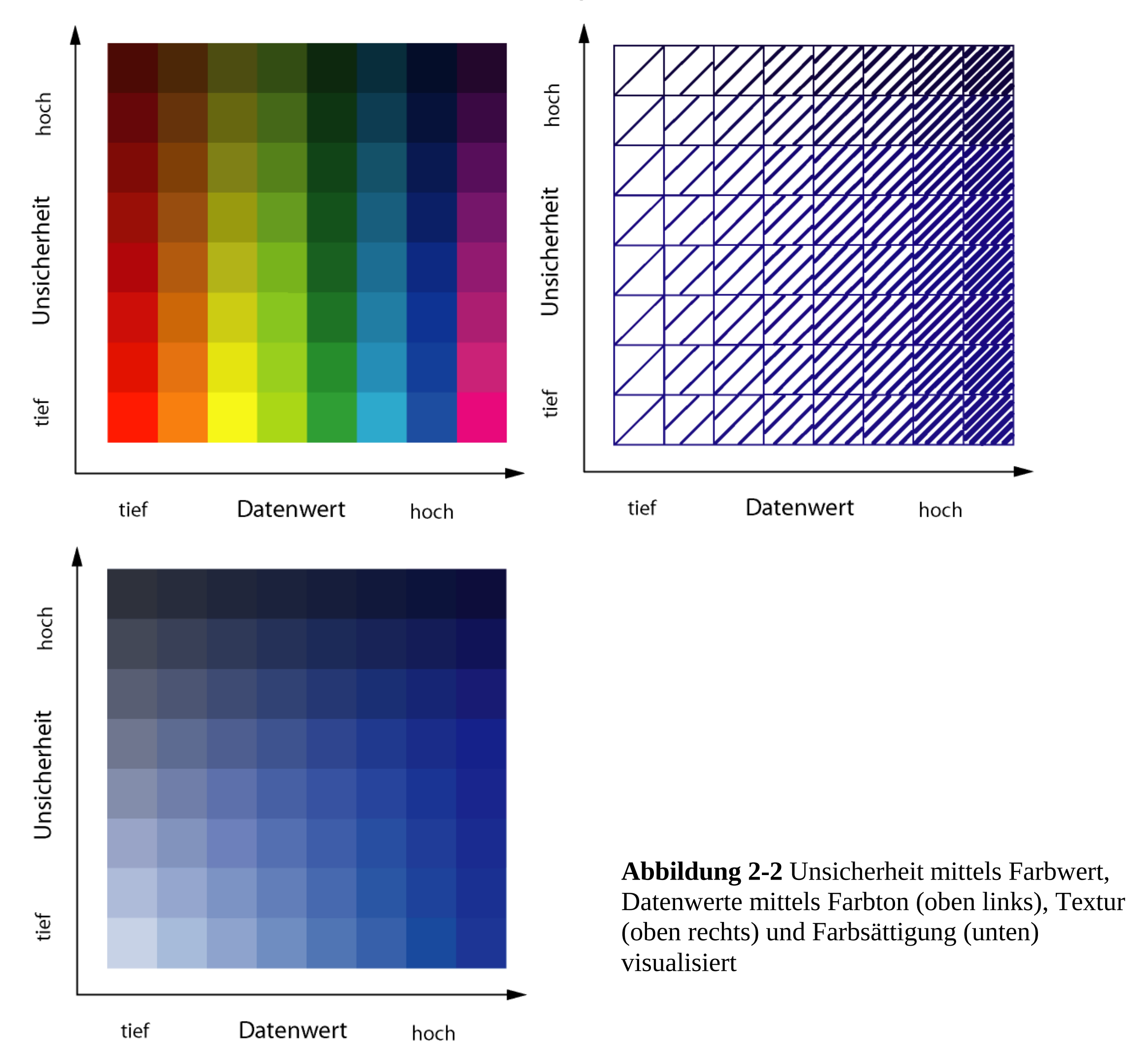


Abbildung 2-2 Unsicherheit mittels Farbwert, Datenwerte mittels Farbton (oben links), Textur (oben rechts) und Farbsättigung (unten) visualisiert

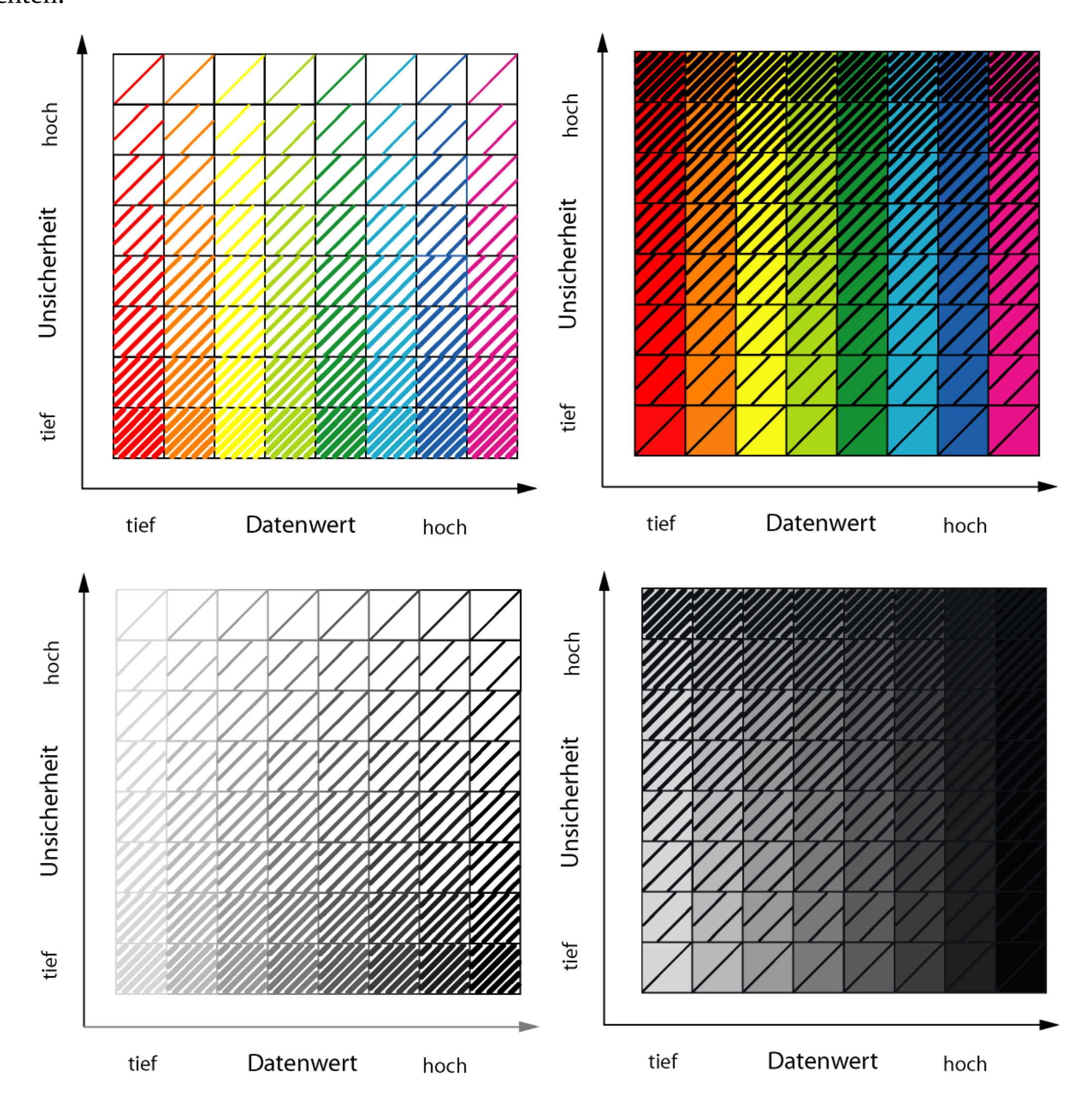
Texture (Textur / Körnung)

Diese Grafikvariable verlangt eine differenzierte Betrachtung von Vektordaten und Rasterdaten. Bei Vektordaten ist es grundsätzlich möglich, die Textur als Gradmesser für die Unsicherheit zu verwenden. Vor allem grosse Flächen lassen viel Spielraum für eingewobene Texturvariationen offen. Bei Rasterdaten ist der Gebrauch auf Zellenbasis sehr schwierig, jede Einzelne davon muss genügend gross dargestellt sein, um darin eine deutlich sichtbare und differenzierbare Textur zu integrieren. Einfacher anwendbar ist die Grafikvariable, wenn man Rasterzellen zu kleinen Unterflächen aggregiert. So bieten diese schon bei kleinen Zellengrössen mehr Zeichnungsfläche. Andererseits entsteht dadurch das Problem des geschickten Zusammenzugs von Zellen. Dieser Verarbeitungsschritt

kann selber wieder Unsicherheiten generieren, oder zumindest Ungenauigkeiten den Daten beimischen. Die Textur kann auf zweierlei Art als optisches Unsicherheitsmass eingesetzt werden: Entweder sie wird in einem zweiten Layer über die eigentlichen Daten gelegt oder sie wird direkt mit diesen kombiniert. Die Textur soll dabei umso stärker ausfallen, je grösser die Unsicherheiten sind. Im ersten Fall werden die darunter liegenden Daten mehr abgedeckt und sind weniger gut sichtbar, was unsicherer bedeutet. Im zweiten Fall wird das Raster umso grobtexturierter, je grösser die Unsicherheit wird. Der Effekt gleicht so einem Rauschen in den Daten oder einer variierenden Generalisierung (Abbildung 2-3).

Caivano (1990, in MacEachren 1995) unterteilt die Textur in drei Aspekte: Bündelung (directionality) umschreibt den Bezug der Texturelemente zueinander, Dichte (density) deren Verteilung bzw. deren Auftretensfrequenz und Grösse (size) bezieht sich auf die Elementgrösse. Veränderungen an einem der drei Aspekte verwandelt die Erscheinung der gesamten Textur. Wie sich dies genau auswirkt, wird hier nicht näher untersucht.

Selbstverständlich ist die Textur gut geeignet, nur die Unsicherheitsdaten darzustellen. Ist der Einsatz von Farben nicht erwünscht oder unmöglich, ist die Datendifferenzierung mittels Texturierung eine beliebte und brauchbare Alternative. Bei nominalen oder ordinalen Daten geht dies problemlos ohne Einschränkungen, bei interval und rational skalierten Daten ist auf eine klare hierarchische Wirkung zu achten.



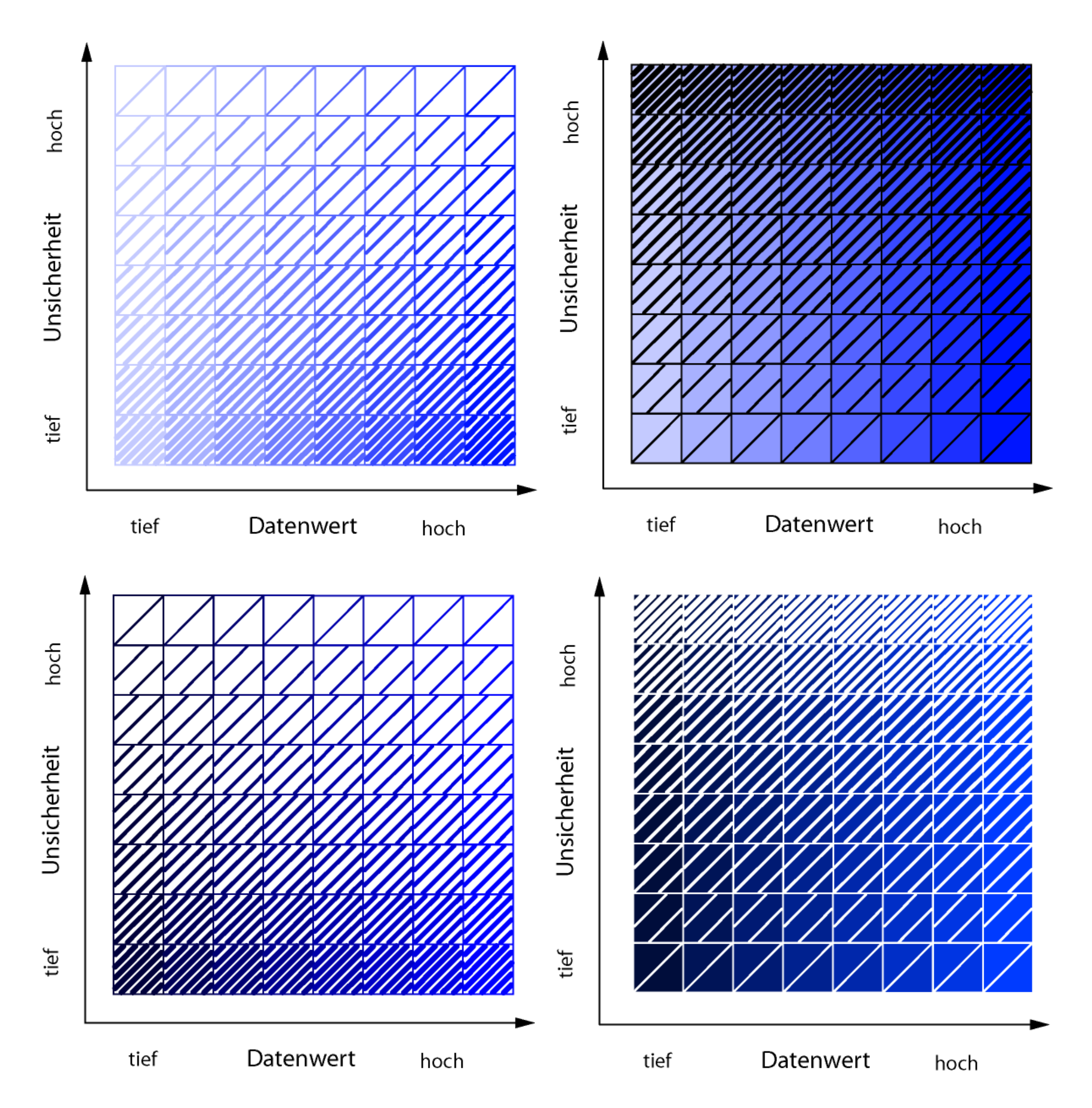


Abbildung 2-3 Unsicherheit mittels Textur, Datenwerte mittels Farbton (Erste Zeile), Grauwert (Zweite Zeile), Farbsättigung (Dritte Zeile) und Farbwert (Vierte Zeile) visualisiert

Size (Grösse)

Der Autor hat für die Grafikvariable Grösse als Stilmittel der Unsicherheitsvisualisierung von Rasterdaten eine Einsatzmöglichkeit entdeckt. Diese ist jedoch eher eine theoretische Spielerei als praktisch umsetzbar. Für Rasterdaten lässt sich in Visualisierungen die Grösse jeder Rasterzelle durch vorgängige Vektorisierung variieren. Umso kleiner die Zelle gezeichnet wird, desto grösser ist die damit symbolisierte Unsicherheit. Die Schrittweite zwischen den Rasterzellen muss dabei erhalten bleiben. Allerdings bieten Computerbildschirme nur wenig Variationsspielraum aufgrund ihrer limitierten Auflösung. Wird das Raster jedoch vektorisiert und die Visualisierung anschliessend ausgedruckt, können viele feine Unterschiede verwirklicht werden. Es bleibt zu überlegen, ob die visuelle Wirkung wirklich noch durch die Grösse als Grafikvariable entsteht, oder nicht eher eine Textur, oder Körnung, erzeugt wird. Darauf gibt es keine objektiv eindeutige Antwort. Je nach Blickwinkel sind beide Grafikvariablen involviert. Das Beispiel zeigt aber auf, dass die Grafikvariablen nicht immer klar separierbar sind, sondern ihre Identifizierung teilweise der subjektiven Wahrnehmung des Betrachters unterliegen. Dies zeigt sich auch bei den nach Bertin eingeführten Erweiterungen.

Bei den Vektordaten ist die vorgeschlagene Skalierung der Bildelemente schwieriger umsetzbar. Einerseits, weil die dargestellten Objekte schon von selbst diverse Grössen mitbringen. Prozentuale Grössenvariationen beeinflussen deshalb nicht alle Bildelemente gleich stark, bei kleineren wirkt sich die Veränderung optisch weniger heftig aus als bei ganz grossen. Dies verzerrt die Wahrnehmung der Unsicherheiten. Andererseits ist es generell schwierig zu bestimmen, wie die Skalierungen vorgenommen werden müssen. Wie sollen Punktdaten verkleinert werden? Sollen Flächen prozentual zu ihrem Inhalt oder zu ihrem grössten Durchmesser schrumpfen? Was geschieht bei mehreren Layern? Dies sind nur drei Fragen eines ganzen sich öffnenden Forschungsfeldes. Buttenfield (1991) wählt die Grösse als mögliche Grafikvariable zur Darstellung von Unsicherheiten für Punkt- und Liniendarstellungen. Clarke und Teague (1999) beziehen auch flächenhafte Darstellungen mit ein. In beiden Artikeln wird die Erklärung für ein praktische Umsetzung weggelassen. Der Autor kann sich deshalb den Einsatz nicht vorstellen (Abbildung 2-4).

Bei den Einzeldarstellung ohne Unsicherheit kann die Grösse nur in Vektordaten verwendet werden. Sie steht da je nach Einsatzgebiet für die massstabsgetreue Abbildung geografischer Räume oder für die gewichtete Repräsentation klassierter Phänomene. Mittels Überzeichnung eines Elementes gegenüber den Restlichen erfährt dieses eine Betonung. Wie sich der Einsatz bei der Darstellung von Unsicherheiten separat auswirkt, bleibt zu testen, ist grundsätzlich aber vorstellbar.

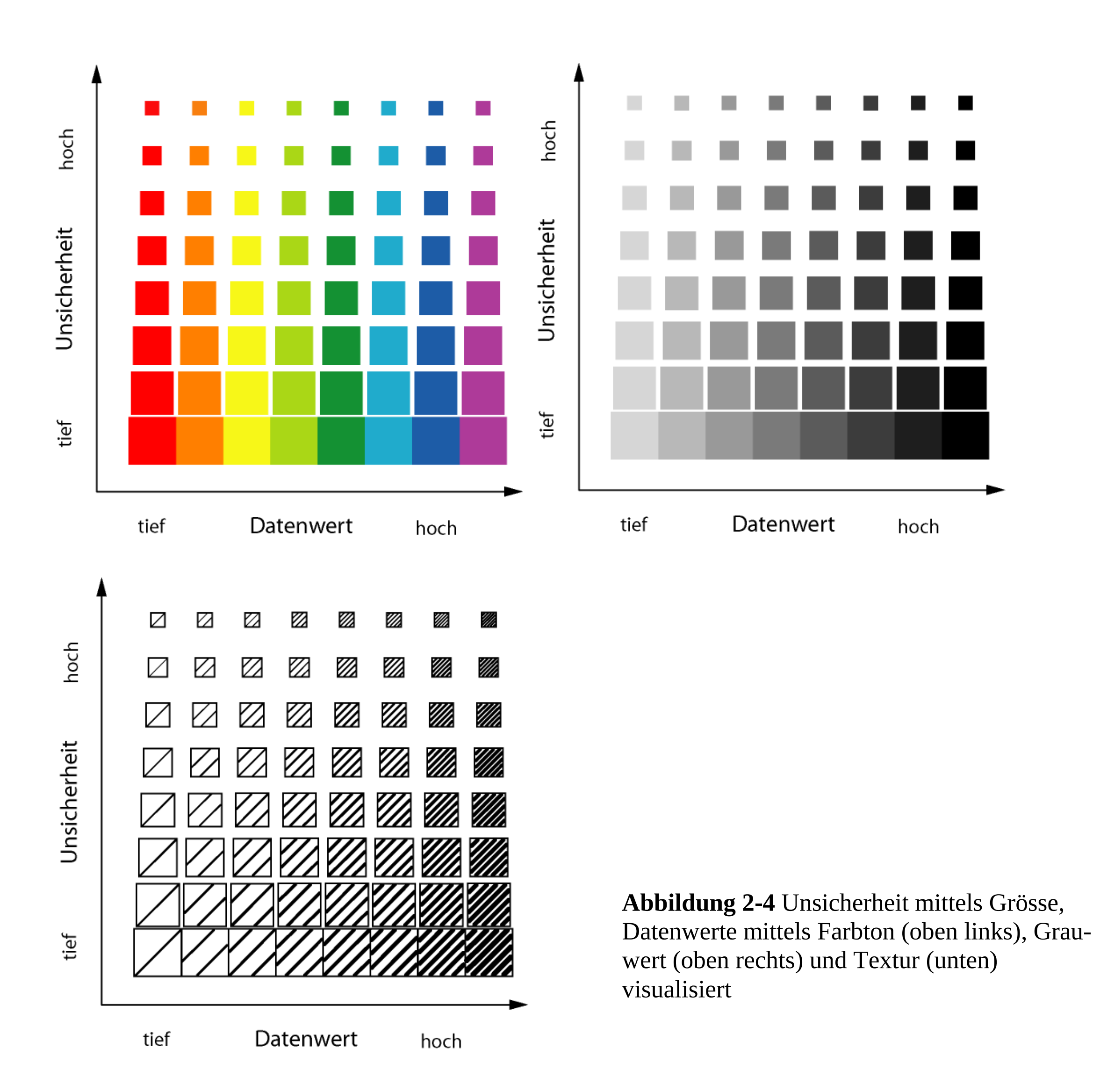


Abbildung 2-4 Unsicherheit mittels Grösse, Datenwerte mittels Farbton (oben links), Grauwert (oben rechts) und Textur (unten) visualisiert

Shape (Form)

Wie bei der Grösse gibt es die theoretische Möglichkeit, die Form zur Unsicherheitsvisualisierung einzusetzen. Folgende Anwendung wurde vom Autor überlegt: Rasterzellen mit 100% gesichertem Inhalt werden quadratisch dargestellt. Mit abnehmender Datenqualität wird die Zelle zunehmend runder gezeichnet. Andere Formen wie Sterne, Striche oder dergleichen wären auch denkbar. Statt eines Formenüberganges könnten auch unterschiedlichen Wahrscheinlichkeitsbereichen verschiedene Symbole zugeordnet werden. Zum Beispiel Haken = >90%, Stern = 80-90%, Mond 70-80%... Die Zellengrösse ist ebenfalls ein limitierender Faktor. Die Visualisierung muss wie schon bei der Textur über kleine Unterflächen aggregiert werden. Dies ist nur bei einem sehr kleinen Raster, dessen Zellen auf dem Bildschirm mit vielen Pixeln gezeichnet werden, nicht notwendig. Die Aussagekraft der resultierenden Visualisierungen wurde nicht getestet. Die Darstellungen sind allerdings weder intuitiv noch übersichtlich. Auf kleinem Raum müssen feine Formenunterschiede erkannt werden. Damit finden sich genügend Gründe, um diese Variable als für die Unsicherheitsvisualisierung ungeeignet zu klassieren (Abbildung 2-5).

Buttenfield (1991) sowie Clarke und Teague (1999) zählen auch die Form zu den einsetzbaren Grafikvariablen. Wiederum fehlt der Beschrieb einer praktischen Anwendung dieser Klassierung. Der Autor kann sich deshalb den Einsatz nicht vorstellen.

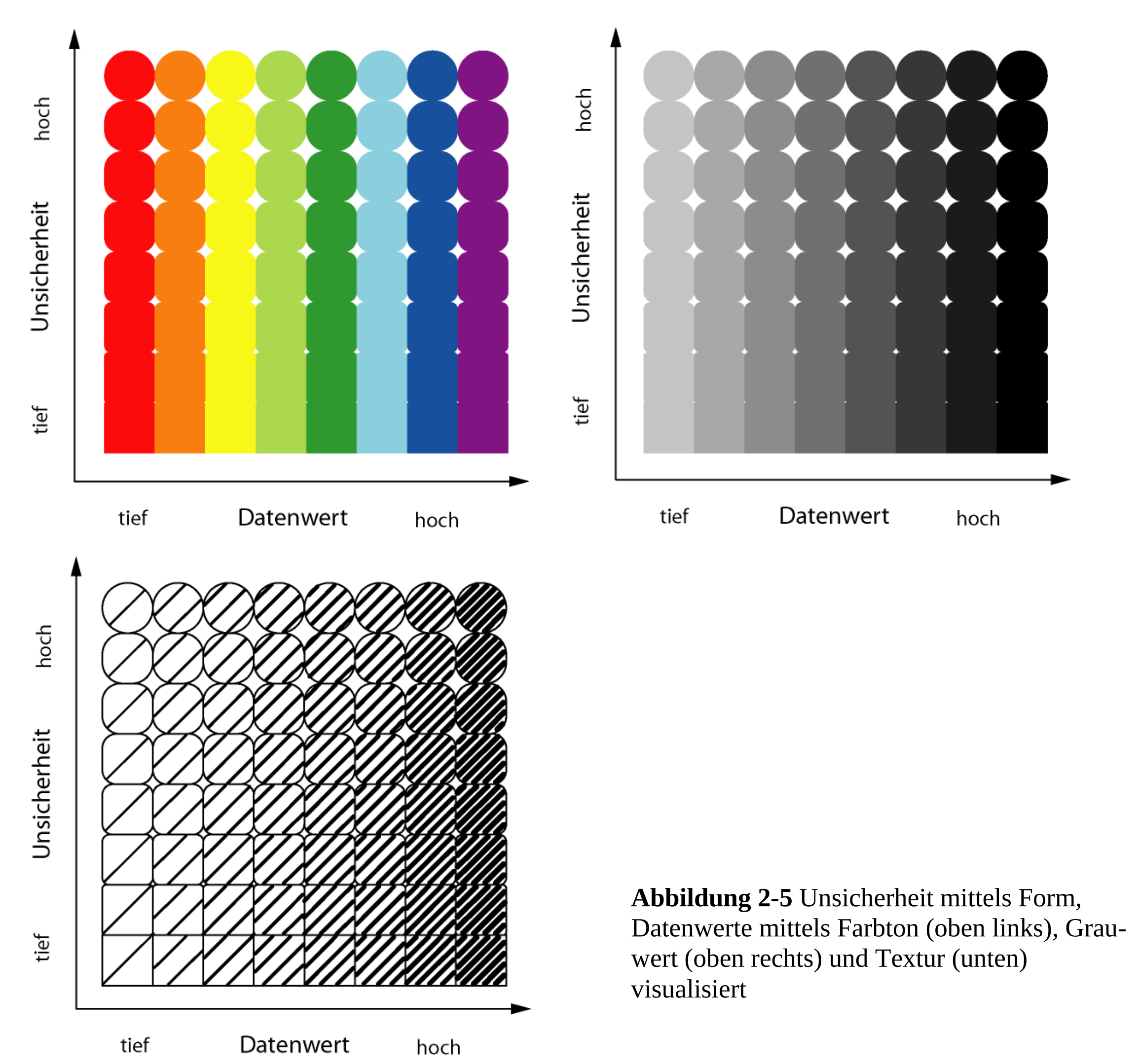


Abbildung 2-5 Unsicherheit mittels Form, Datenwerte mittels Farbton (oben links), Grauwert (oben rechts) und Textur (unten) visualisiert

Orientation (Orientierung)

Auch für die Orientierung gibt es technisch realisierbare Möglichkeiten. Wiederum eine Idee des Autors sind beispielsweise Orientierungspfeile, die je nach Unsicherheit in eine Richtung rotiert werden. Der Nordpfeil wird der 100% Sicherheit zugeordnet, und mit Drehung in Uhrzeigerrichtung nimmt die Sicherheit ab. Diese Lösung bietet die gleichen Nachteile wie jene für die Form: Sie lässt sich nur über aggregierte Rasterzellen anwenden oder jede einzelne muss relativ gross gezeichnet werden. Für die Orientierung gilt demnach das Selbe wie für die Form: Der Einsatz der Variable ist für den untersuchten Zweck ungeeignet (Abbildung 2-6).

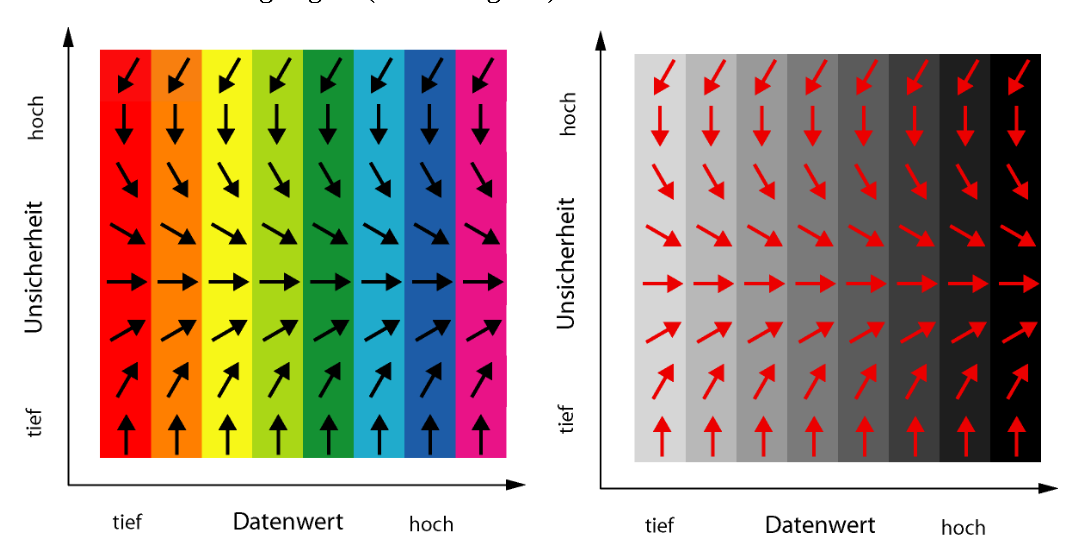


Abbildung 2-6 Unsicherheit mittels Orientierung, Datenwerte mittels Farbton (links) und Grauwert (rechts) visualisiert

Location (Platzierung)

In manchen Artikeln wird die räumliche Platzierung als siebte Grafikvariable unterschlagen, da sie eine Sonderstellung besetzt. Für alle naturräumlichen Daten ist ihre Ausprägung fixiert. Eine frei gewählte Zuordnung verfälscht die Daten nachhaltig und führt Benutzer dieser Visualisierungen in die Irre. In der modernen Geographie werden jedoch auch kartenähnliche Abbildungen aus Daten ohne naturräumlichen Bezug produziert, die für diese Variable eine grössere Wahlmöglichkeit offerieren. Es wäre demnach nachlässig, die Platzierung als Grafikvariable bei Seite zu lassen. Raster geben die Platzierung jedoch vor, nur der Attributwert der Zellen ändert. Für die in dieser Arbeit besprochenen Eisschildmodellierungen findet der Autor aufgrund obiger Aussage keine sinnvolle Anwendung.

2.4.3 Erweiterte Grafikvariablen

In den 60er Jahren war die gedruckte Karte noch die übliche Form geographischer Darstellungen, die Bertin massgeblich beeinflussten. Seit der Veröffentlichung des Regelwerkes durch Bertin wird nach Erweiterungen desselben geforscht. Der Anwendungsbereich der Geovisualisierungen hat sich stetig erweitert. Die neuen Medien haben dazu beigetragen, dass die ursprünglichen Grafikvariablen nicht mehr ausreichen, um alle grafischen Komponenten präzise zu beschreiben. Wie im vorhergehenden Abschnitt folgt nun eine Beschreibung der einzelnen neueren Grafikvariablen und eine Untersuchung zu ihrem Einsatz in der Unsicherheitsvisualisierung.

Color Saturation (Farbsättigung)

Morrison (1974, in Blok 2000) schlug als erste Ergänzung der Grafikvariablen die Farbsättigung vor. Obwohl Bertin sie bereits erwähnte, trennte er sie nicht vom Farbwert. Die Farbsättigung ist hervorragend geeignet für den Einsatz in der Unsicherheitsvisualisierung. Für MacEachren ist sie sogar die logischste aller Grafikvariablen zur Anzeige von Unsicherheiten (1992, in van der Wel et al. 1994). Buttenfield (1991) schlägt sie zur kontinuierlichen Darstellung der Positionsgenauigkeit vor. Der Effekt der Farbsättigung lässt sich mit jenem des Farbwertes vergleichen, wirkt sich dabei aber genau umgekehrt aus. Die Farbsättigung wird mit einem Wert zwischen 0 und 1 angegeben, wobei 1 den vollen Farbton bezeichnet, während 0 keine Zugabe von Farbe bedeutet, das betroffene Bildelement also weiss erscheint. Somit lassen sich schöne Übergänge von intensiven Farben über immer blassere zu schliesslich ganz weissen Flächen erzeugen. In diese Richtung nimmt die Unsicherheit des gezeichneten Wertes zu und dessen Wahrscheinlichkeit ab. Ein Betrachter kann den Effekt rasch erkennen und ihn intuitiv richtig zuordnen. Das Verblassen der Bildelemente ist leicht mit der sich verschlechternden Datenqualität in Verbindung zu bringen. In Kombination mit dem Farbwert ergeben sich sehr schöne Darstellungen. Allerdings ist die Farbsättigung, wie der Farbwert, nicht mit einer Graustufendarstellung zusammen einsetzbar, denn sie wirkt ebenfalls auf die Helligkeit ein. In Verbindung mit durch Texturen gezeichneten Ursprungswerten ist ein Einsatz gut möglich (Abbildung 2-7).

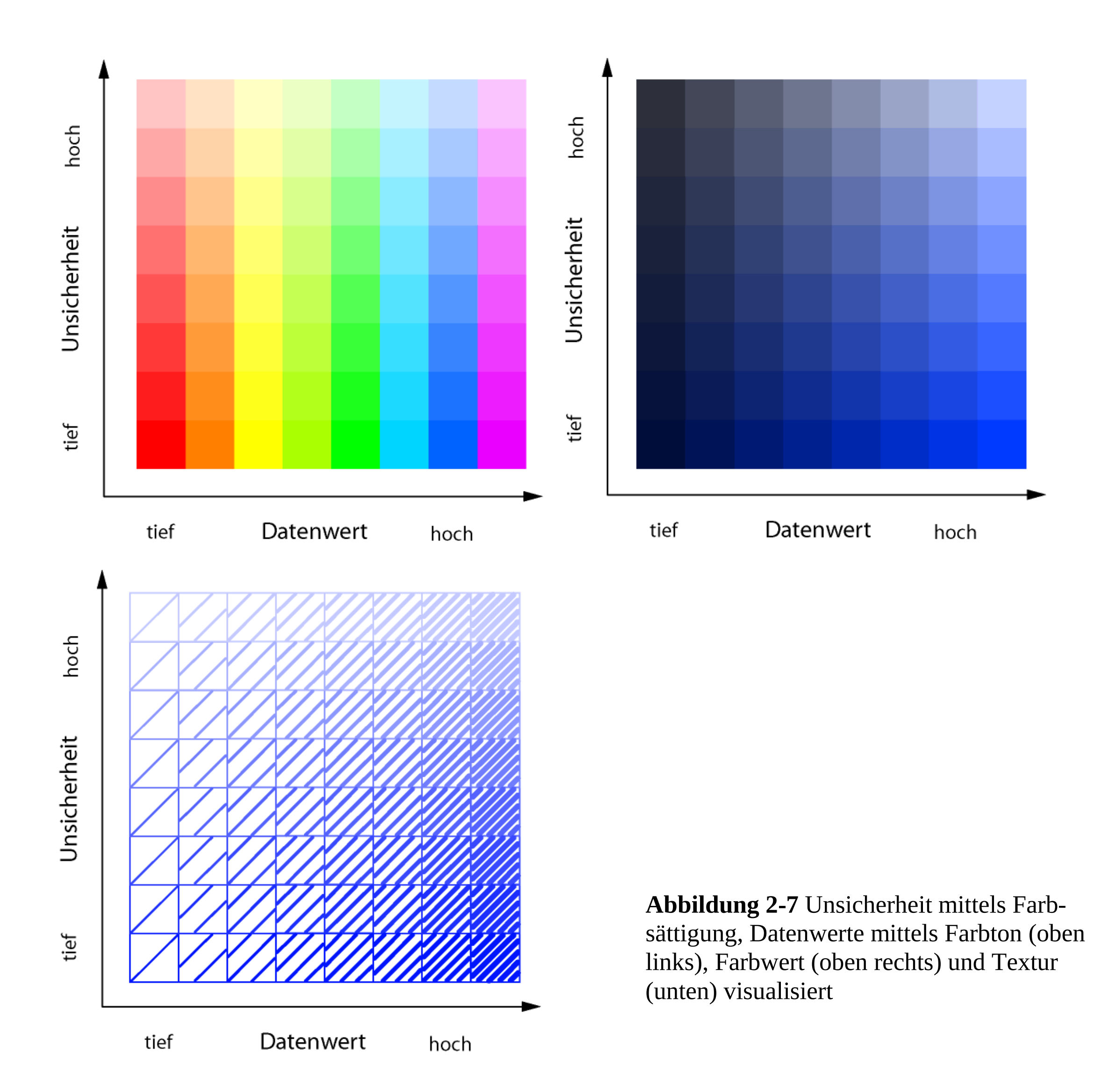


Abbildung 2-7 Unsicherheit mittels Farbsättigung, Datenwerte mittels Farbton (oben links), Farbwert (oben rechts) und Textur (unten) visualisiert

Pattern (Muster)

Diese und auch die folgende Grafikvariable gehören einer höheren Abstraktionsstufe an. MacEachren (1995) erwähnte das Muster als mögliche Einteilung höherer Stufe, da es aus den Bertin'schen Grundvariablen Form, Orientierung, Textur und Grösse, sowie der Gliederung (arrangement) als von Morrison (1974, in Blok 2000) vorgeschlagene Grafikvariable besteht. Das Muster kann zusammen mit dem Farbton, dem Farbwert und der Farbsättigung verwendet werden, allerdings muss auf die selben Einschränkungen wie bei der Textur hingewiesen werden, mit der es viele Ähnlichkeiten aufweist. Es konnte keine konkrete Anwendungsmöglichkeit entdeckt werden, weshalb hier nicht weiter auf das Muster eingegangen wird.

Clarity / Focus (Klarheit)

Dies ist eine weitere komplexe Grafikvariable, die 1992 als Fokus von MacEachren vorgestellt und 1995 als Klarheit von ihm weiterentwickelt wurde (Blok 2000). Sie beschreibt die Schärfe oder den Grad an Verzerrung, der ein Bildelement unterworfen ist. Drei Unterkomponenten definieren sie: Die (Kanten)schärfe (crispness), die Auflösung (oder Abstrahierungsgrad) und die Transparenz. Jede Komponente kann separat auf ihre Eignung zur Unsicherheitsvisualisierung untersucht werden. Die Kantenschärfe ist in flächenhaften Darstellungen besonders effizient. So können in einer Bodentypenkarte die fliessenden Übergänge der einzelnen Bodentypen realistisch symbolisiert werden. Anstelle einer scharfen Trennlinie zwischen den beiden Zonen werden sie ineinander überblendet. In Rasterdaten ist dies ein konzeptionelles Problem, da im Gegensatz zu Vektordaten keine topologischen Daten vorhanden sind. Es müsste zuerst definiert werden, mit welchen anderen Pixeln die Mischung stattfinden soll. Die Integration des Abstrahierungsgrades ist in Rasterdaten wiederum einfacher zu bewerkstelligen, indem dort je nach Unsicherheit mehr oder weniger Rasterzellen miteinander aggregiert werden. Die Transparenz ist die interessanteste Komponente der Klarheitsvariable. Mit ihr lassen sich hervorragende Effekte erzielen. Ein zweiter Layer, über die echten Werte gelegt, erlaubt die Variation der Transparenz. Die darunter liegen Daten scheinen mehr oder weniger stark durch. Je sicherer, desto deutlicher sollten sie erscheinen. Der Effekt gleicht einem Nebel, durch den die Daten durchscheinen. Dies ist eine treffende Metapher zur Datenqualität.

2.4.4 Visualisierungstechniken

Neben dem Einsatz einzelner Grafikvariablen muss auch die Anwendung komplexer oder dynamischer Visualisierungstechniken erwähnt werden. Trotz ihrer ganz unterschiedlichen Herstellung besitzen sie die Gemeinsamkeit, dass ihre Anwendung nicht durch ein blosses Einwirken auf eine grafische Variable beschränkt ist.

Blinken

Diese Technik wird als dynamische Grafikvariable bezeichnet, denn das Blinken wirkt sich auf ein spezifisches Element der Visualisierung aus. Der Autor definiert vier Attribute, die das Blinken bestimmen. Das Erste ist die Frequenz, die die Wiederholrate des Wechsels zwischen zwei Extremzuständen beschreibt. Das Zweite ist die Grösse des blinkenden Bereiches. Das Dritte ist die Intensität des Blinkens. Ein Blinken, das durch einen kleinen Farbübergang wie z.B. von Hellrot zu Dunkelrot erzeugt wird, wirkt schwächer als der Effekt durch einen starken Übergang von z.B. Weiss zu Schwarz. Das vierte Attribut ist die Schärfe das Farbüberganges. Der Wechsel zwischen zwei Farbtönen kann abrupt und direkt gezeichnet werden oder er wird verfeinert mit Farbton-

zwischenstufen, die einen sanfteren Blinkeffekt erzeugen. Im zweiten Fall ist der Wechsel kontinuierlich, während im ersten ein abgehackter Rhythmus besteht. Für die Darstellung der Unsicherheit eignen sich zwei der vier Attribute gut. Einerseits kann die Frequenz eingesetzt werden. Sie wird dort besonders hoch sein, wo die Datenqualität am fragwürdigsten ist. Bei gesicherten Daten wird gar kein Blinken gezeigt. Andererseits kann die Intensität des Blinkens variiert werden, so dass Bereiche mit grösserer Unsicherheit stärkere Farbtonwechsel erfahren.

Die Blinktechnik lässt sich besonders gut mit Schwellenwerten einsetzen. Liegen die Wert über einem gewissen Unsicherheits-Schwellenwert, blinken die dargestellten Datenbereiche, darunter verhalten sie sich statisch. Der Benutzer soll dann mittels einfacher Bedienung den Schwellenwert variieren können, beispielsweise mit einem Schieber (threshold slider). Fisher (1992, in van der Wel et al. 1994: 326) setzt die Blink-Technik ein. Dabei ist die Darstellungsdauer auf dem Bildsschirm umgekehrt proportional zur Unsicherheit der Datenwerte. Die Technik ist für Druckerzeugnisse nicht verwendbar. Das statische Medium Papier lässt keinen Anzeigewechsel zu.

Ungeklärt ist die Art des Farbwechsels. Unterschiedliche Farbgebungen verlangen verschiedene Wechsel. Diese müssen klar definiert werden, was je nach dargestelltem Raster schwierig ist. Ein anderer entscheidender Punkt ist die Benutzerfreundlichkeit. Es ist anzunehmen, dass ein ständig blinkender Bildschirm den Betrachter schnell ermüden lässt, seine Konzentration damit nachlässt. Denkbar wäre sogar das Hervorrufen körperlicher Beschwerden wie beispielsweise Kopfweh. Der Einsatz dieser Visualisierungstechnik sollte sporadisch erfolgen und eher kurze Zeiträume umfassen. Der Autor bezweifelt aber, dass es Anwender gibt, die sich freiwillig längere Zeit dem Blinken aussetzen würden.

3-Dimensionalität

Wird die dritte Dimension in die Visualisierungen miteinbezogen, eröffnet sich ein weiterer Freiheitsgrad zur Unsicherheitsdarstellung. Es kann eine 3-dimensionale Oberfläche mitgezeichnet werden (van der Wel et al. 1994: 325). Mit dem Höhenmodell lassen sich zwei Arten von Darstellungen erzeugen. Entweder die Höhen entsprechen den Daten, dann werden die Unsicherheiten als planare Fläche auf dieses Höhenmodell projiziert. Dies bietet sich natürlich bei echten Höhendaten an, aber auch sonstige räumliche Datensets können so gezeichnet werden. Oder die Daten liegen als 2-dimensionale, flache Karte vor. Dann wird das Höhenmodell als Unsicherheitsoberfläche verwendet. Bei dieser Visualisierungstechnik ist es wichtig, dass die darstellungserzeugende Applikation die Möglichkeit bietet, die Daten aus allen Richtungen zu betrachten. Je nach dargestelltem Blickwinkel können gewisse Datenbereiche durch andere verdeckt sein. Idealerweise geschieht dieser Kamerapositionswechsel in Echtzeit.

Animation

Als weitere Visualisierungsdimension kann die Zeit eingesetzt werden. Durch Aneinanderfügen verschiedener Modelldarstellungen kann eine Animation erzeugt werden. Amrhein (1991) beurteilt diese als schnelle und effiziente Methode, grosse Datenvolumen darzustellen. Liegen 50 Resultatesets aus einer Monte Carlo Simulation vor, kann anstelle eines mathematischen Zusammenzuges dieser Daten und der Berechung der Abweichungen eine Animation aus 50 Bildern kreiert werden. Empfehlenswert dabei ist das Sortieren aller räumlich deckungsgleicher Werte nach ihrer Grösse, um die Animation sanft zu gestalten. Bei unsortierten Werten kann mitunter ein chaotisches Flimmern entstehen. Dieses kann mit längeren Wechselzeiten vermieden werden. Die Technik wird von

manchen Autoren allerdings nicht mehr Animation, sondern Sequenzierung bezeichnet, wenn jede Stufe des Ablaufes separat klar erkennbar ist (Muehrke 1990, in van der Wel et at. 1994: 326). Mit der Sequenzierung können auch verschiedenartige Visualisierungen hintereinander gezeigt werden, um deren unterschiedliche optische Wirkung zu demonstrieren.

Univariate Visualisierungen

Bisher wurden nur Visualisierungen präsentiert, die bivariat Daten und Unsicherheitsmetadaten zusammen darstellen. Jetzt sollen noch kurz univariate Visualisierungen angesprochen werden, die zwei separate Fenster für die Darstellung benötigen. Es wird nicht jede mögliche Grafikvariablenkombination besprochen, sondern bloss das jeweilige Konzept vorgestellt.

Zwei statische Karten

In dieser Technik sind beide Datensätze gleichzeitig in getrennten Fenstern neben- oder übereinander visualisiert. Dies bietet einige Vorteile. Die Darstellungen können ohne einander zu beeinflussen alle Grafikvariablen nutzen. Sie sind klar, verständlich, übersichtlich (je nach Daten) und wirken weniger überladen wie manche bivariate Visualisierung. Nachteile sind ebenfalls in Kauf zu nehmen. Der Platzbedarf für die zwei Fenster ist doppelt so gross. Da die Daten und Unsicherheiten nicht übereinander liegen, sind die zusammengehörenden Werte nicht immer ganz eindeutig identifizierbar.

Wechselweise Anzeige beider Karten

Wie bei den zwei statischen Karten werden die Daten und die Datenqualität getrennt dargestellt. Hier allerdings im selben Fenster alternierend. Der kleinere Platzbedarf ist dabei ein Vorteil, dafür muss sich der Betrachter die jeweils nicht sichtbaren Daten vorstellen bzw. aus dem Gedächtnis abrufen. Ein sehr schnelles Wechseln zwischen den beiden Ansichten wäre interessant zu testen. Für den Betrachter könnte ein Mischeffekt entstehen, sofern der Wechsel schneller ist als die Reaktionszeit seiner Augen. Dies würde ca. 15-20 Wechsel pro Sekunde erfordern. Evans (1996) verwendet eine Endlosschleife, in der zwei unterschiedliche Zustandskarten alternieren. Sie benutzt dabei eine Änderungsgeschwindigkeit von vier Wechseln pro Sekunde.

2.4.5 Weitere Techniken und Effekte

In der Literatur sind weitere Visualisierungstechniken zu finden. Diese sind jedoch für diese Diplomarbeit weniger relevant, oder sie entstehen durch Verbindung mehrerer bereits vorgestellter Methoden. Bei manchen erscheint dem Autor das Konzept nicht sehr vielversprechend, wiederum andere unterscheiden sich nebst der Benennung kaum. Deshalb hier nur eine kurze Erwähnung aller noch gefundener Techniken:

Zooming (Optische Formatänderung)

Die Anwendung unterschiedlicher Zoomstufen – bzw. unterschiedlichen Detailierungsgrad – zur Veranschaulichung der Unsicherheit wird von van der Wel et al. vorgeschlagen (1994: 325). Je grössere Unsicherheiten in den Daten enthalten sind, desto weniger Detail werden dargestellt, desto stärkere Generalisierung wird auf die Visualisierung angewendet. MacEachren (1992) zeigt mit der Auflösung eine identische Lösung unter anderem Namen.

Slicing (In Teile trennen)

Bei dieser Technik werden zeitlich versetzt unterschiedliche Wahrscheinlichkeits-Schwellenwerte bestimmt und die Daten einer der beiden Teilmengen zugeordnet und visualisiert. Mittels Veränderung des Schwellenwertes färben sich unterschiedliche Gebiete ein (van der Wel et al. 1994: 325).

Shading (Schattierung)

Um in 2-dimensionalen Anzeigen eine Unsicherheitsoberfläche zu simulieren, kann diese Schattierungstechnik eingesetzt werden. Nachteil daran sind künstliche Schatteneffekte in Bereichen guter Datenqualität ‚hinter Unsicherheitsbergen' (van der Wel et al. 1994: 325).

Dazzling (Optische Verwirrung)

Beim Dazzling soll eine unangenehme Musterung den Betrachter von weniger wahrscheinlichen Datenbereichen ablenken. Van der Wel et al. (1994: 327) schlagen dafür unter anderem Interferenzmuster vor.

Soundeffekte

Eine sehr interessante Methode, um die visuellen Komponenten für andere Dienste frei zu halten, ist der Einsatz von Soundeffekten (Krygier 1994, Fisher 1994, in van der Wel et al. 1994). Während der Benutzer der Visualisierung mit dem Mauszeiger über die Darstellung fährt, kann ein Ton mit unterschiedlicher Modulation produziert werden. Entweder zeigt die Lautstärke die sich ändernde Unsicherheit an, oder die Tonlage wird dafür eingesetzt. Denkbar ist dieses Mittel für Personen mit Sehschwächen, bei überladenen Visualisierungen oder bei schlecht erkennbaren Unterschieden. Die Anwendung bleibt auf computergestützte Darstellungen beschränkt.

Geruchseffekte

Tatsächlich wird an der Erzeugung künstlicher Gerüche geforscht. Funktioniert diese Technik dereinst, ist ein Einsatz als Unsicherheitsanzeiger denkbar. Ein Konzept des Autors ist, unsicheren Datenbereiche einen üblen Geruch zuzuordnen. Die Nutzungsbereitschaft der Anwender bleibt zu testen. Als Warnsignal ist dieser Effekt bestimmt effizient, da er nur schwer ignorierbar ist.

2.4.6 Zusammenfassung der Visualisierungsoptionen

In der folgenden Tabelle 2-1 werden alle beschriebenen Grafikvariablen und Visualisierungstechniken aufgelistet. Die erste Spalte gibt den Namen der bewerteten Visualisierungsoption an (es sind damit die Grafikvariablen und die Visualisierungstechniken gemeint). Die zweite Spalte (univariat) bewertet den möglichen Einsatz für die univariate Darstellung der Unsicherheit (also nur die Datendifferenzen). Analog dazu gibt Spalte drei die Bewertung für den bivariaten Einsatz (Daten und Metadaten zusammen) an. Die unterteilte Spalte vier schliesslich zeigt an, ob die Technik in der anschliessend beschriebenen Umsetzung implementiert ist. Dabei gilt der erste Wert für den univariaten, der zweite für den bivariaten Einsatz. Die Tabelle bezieht sich ausschliesslich auf Rasterdaten.

			Implementiert	
Bezeichnung	**Univariat**	**Bivariat**	**Univariat**	**Bivariat**
Farbton	✓	Theoretisch	✓	Nein
Farbwert	✓	✓	✓	✓
Textur	✓	✓	Nein	✓
Grösse	Nein	Theoretisch	Nein	Nein
Form	Nein	Theoretisch	Nein	Nein
Orientierung	Nein	Theoretisch	Nein	Nein
Platzierung	Nein	Nein	Nein	Nein
Farbsättigung	✓	✓	✓	✓
Muster	✓	✓	Nein	Nein
Klarheit (Transparenz)	Nein	✓	Nein	Nein
Blinken	Nein	✓	Nein	Nein
3-Dimensionalität	✓	✓	Nein	Nein
Animation	✓	✓	(✓)	(✓)
Zwei statische Karten	---	✓	---	✓
Wechselweise Anzeige	---	✓	---	Nein
Zooming	Nein	✓	Nein	Nein
Slicing	Nein	✓	Nein	Nein
Schattierung	Nein	✓	Nein	Nein
Dazzling	Nein	✓	Nein	Nein
Soundeffekte	Nein	✓	Nein	Nein
Geruchseffekte	Nein	✓	Nein	Nein

Zeichenerklärung

✓	=	Der Einsatz der Visualisierungsoption ist uneingeschränkt möglich.
Theoretisch	=	Für die Visualisierungsoption besteht ein theoretisches Konzept. Der Nutzen einer praktischen Umsetzung ist zweifelhaft.
Nein	=	Die Verwendung der Visualisierungsoption ist unmöglich
---	=	Es kann keine Aussage gemacht werden. Die Techniken sind grundsätzlich auf mehrere Darstellungen beschränkt.
(✓)	=	Die Visualisierungsoption wird nur in eingeschränktem Umfang implementiert.

Tabelle 2-1 Zusammenfassung der besprochenen Grafikvariablen und Visualisierungstechniken

3 Daten

Dieses Kapitel beschreibt die Beispieldaten, die für den Praxistest der entwickelten Applikation zur Visualisierung von Unsicherheiten verwendet werden. Diese Daten beschreiben Ergebnisse einer Eisschildmodellierung, in der das Edinburgh Ice Sheet Model (EIS) eingesetzt wird (Hagdorn 2003).

3.1 Eisschildmodellierung

Die Beispieldaten entstammen einer Eisschildmodellierung (ISM für Ice Sheet Model). Die Berechnung dieser Modelle geschieht zumeist sehr grossskalig, das heisst in kontinentalen oder gar globalen Massstäben. Das führt zu einem recht hohen Abstraktionsgrad, der eine Reihe von Unsicherheitsquellen mit sich bringt. Die Grösse einer Rasterzelle beträgt in den Beispieldaten 20 km. Diese Rasterzellen wurden aus einem feiner aufgelösten Höhenmodell mit rund 100 m Rasterweite generalisiert. Wie in Kapitel 2 ‚Grundlagen' beschrieben, sind Generalisierungen eine wichtige Unsicherheitsquelle. In diesem Fall beeinflusst sie entscheidende Modellparameter wie minimale und maximale Geländehöhe, Hangneigung und Rauhigkeit des Untersuchungsgebietes. Diese Parameter steuern das ISM insofern, dass die Temperatur und der Niederschlag von diesen orographischen Effekten abhängen, und somit das Mikro- wie das Makroklima prägen. Schliesslich wirken sich diese Faktoren auf die Akkumulations- und Ablationsgebiete eines Gletscher/Eisschildes aus, die wiederum bestimmen, ob sich die untersuchte Eismasse ausdehnt oder zurückzieht. Der Modellierungszeitraum umfasst 120'000 Jahre. Für jeweils 1'000 Jahre wird der berechnete Systemzustand in eine Datei geschrieben. Somit entstehen 121 Zeitschritte (Hebeler 2005).

Das modellierte Gebiet entspricht in etwa Fennoskandinavien. Dieses Gebiet umfasst die drei Länder Norwegen, Schweden und Finnland. Die Topographie dieser Region ist sehr abwechslungsreich: Im Westen an der norwegischen Atlantikküste graben sich tiefe Fjorde kilometerweit ins Landesinnere, um in eine gebirgige Landschaft überzugehen. Weiter ostwärts verflacht das Gebiet zunehmend. Finnland schiesslich ist geprägt durch weite Ebenen mit unzähligen Seen.

Die Modellierung vergangener Eisschilde, wie hier jene des Letzen Glazialen Eishöchststandes (LGM für Last Glacial Maximum) über Fennoskandinavien, soll bestehende Modelle anpassen und verbessern helfen, um damit Berechnungen unter zukünftigen Klimamodellen zu ermöglichen. Dies ist vor allem wichtig bei der Abschätzung der Reaktion bestehender grosser Eismassen wie der Antarktis, Grönland oder Patagonien auf den Einfluss von Klimaveränderungen und den damit verbundenen Meeresspiegelschwankungen.

Die Unsicherheiten (beziehungsweise Differenzen) in den verwendeten Daten wurden durch Variation der Topographie und den daraus folgenden Änderungen der Temperatur erzeugt. Die Temperatur ist dabei abhängig vom adiabatischen Gradienten. Ihre Veränderung beträgt rund 0.7 Grad Celsius je 100 m. Die Erhöhung respektive Vertiefung der topografischen Höhe über Normalnull erfolgte mit konstanten oder variablen Beträgen. Insgesamt wurden 12 verschiedene Modellierungsresultate erstellt, die zur Erprobung der unterschiedlichen Visualisierungen zur Verfügung stehen. Folgende Höhenunterschiede wurden in die Ausgangstopographie eingerechnet (Tabelle 3-1).

Datei	**Topographische Änderung**
run1-2.1000.20km.nc	Originaltopographie
top_a100.1000.20km.nc	Konstant 50m Erhöhung über ganzes Gebiet addiert
top_a250.1000.20km.nc	Konstant 250m Erhöhung über ganzes Gebiet addiert
top_a50.1000.20km.nc	Konstant 100m Erhöhung über ganzes Gebiet addiert
top_aperc10.1000.20km.nc	Konstant 10% Erhöhung über ganzes Gebiet addiert
top_apercrand10.1000.20km.nc	Zufällig bis max. 10% Erhöhung über ganzes Gebiet addiert
top_arand100.1000.20km.nc	Zufällig bis max. 100m Erhöhung über ganzes Gebiet addiert
top_arand250.1000.20km.nc	Zufällig bis max. 250m Erhöhung über ganzes Gebiet addiert
top_cr10.1000.20km.nc	Zufällig bis max. 10m Änderung über ganzes Gebiet (+ oder -)
top_cr100.1000.20km.nc	Zufällig bis max. 100m Änderung über ganzes Gebiet (+ oder -)
top_subtrand100.1000.20km.nc	Zufällig bis max. 100m Vertiefung über ganzes Gebiet subtrahiert
top_subtrand250.1000.20km.nc	Zufällig bis max. 250m Vertiefung über ganzes Gebiet subtrahiert

Tabelle 3-1 Die zwölf Beispieldatensätze und ihre topographischen Unterschiede

3.2 NetCDF Datenformat

NetCDF (network Common Data Form) (Unidata 2005) ist sowohl eine array-orientierte Schnittstelle als auch eine Bibliothek, die einen schnellen und direkten Zugriff auf die gespeicherten Daten erlaubt. Das Format wird in verschiedenen Forschungszweigen zur persistenten Speicherung von Modellierungsdaten verwendet, zum Beispiel bei Berechnungen von Klimamodellen. Die beschriebenen Daten der Modellierung des Nordeuropäischen Eisschildes liegen im NetCDF Datenformat vor. Für den Test der Applikation mit diesen realen Modellierungsdaten ist eine Anpassung der Datenschnittstelle an dieses Datenformat implementiert. Interessant sind NetCDF Datensätze aus mehreren Gründen. Einerseits können verschiedene Modellvariablen in ein und derselben Datei abgelegt werden, andererseits können eine grosse Zahl von Datendimensionen definiert werden, die den Datenraum der Variablen aufspannen. Eine Dimension kann offen gelassen werden, das heisst deren Grösse (Anzahl Schritte) muss nicht von Beginn der Modellkalkulation bekannt sein. So kann in deren Verlauf die Datei mit den neu berechneten Daten in diese Dimension ergänzt, sogenannte ‚slices' hinzugefügt werden. Bei den verwendeten Versuchsdaten ist diese Dimension die Zeit, weshalb von ‚time slices' (Zeitscheiben) gesprochen wird.

3.3 Aufbau der Dateien

Mit der NetCDF Bibliothek werden Hilfsprogramme mitgeliefert, die eine Transformation vom NetCDF Binärformat in ASCII Dateien und zurück erlauben. So kann die Struktur und der Inhalt der Dateien in einfachen Texteditoren betrachtet werden. NetCDF Dateien bestehen aus zwei Teilen: Dem Dateikopf und dem Dateikörper.

Im Dateikopf werden die gespeicherten Informationen definiert und beschrieben. In der ersten Zeile steht das Dateiformat (netcdf) und der Name der Datei. Anschliessend werden die verwendeten Dimensionen benannt und begrenzt. Eine Dimension kann mittels UNLIMITED offen gelassen werden. Nun können die Variablen definiert werden. Als erstes müssen die Dimensionen erläutert werden, denn jede Dimension gilt auch als Variable. Jeder Variable wird ein Typ zugeordnet, ein Kurzname vergeben und die begrenzenden Dimensionen angegeben. Ist eine Variable auch eine Dimension, wird sie durch sich selber begrenzt. Zu jeder Variablendefinition können weitere Attribute angefügt werden. In den benutzten Eisschilddateien sind dies ein Langname und die Grösseneinheiten (Meter, Grad Celsius, und weiter). Im Anschluss an den Variablenblock können globale Attribute (Konstanten) zugewiesen werden. Die Dimensionen und Variablen der benutzten Eisschild-Modellierungsdaten werden in der unten folgenden Tabelle aufgelistet und beschrieben. Fettschrift zeigt an, dass es sich bei der Variable um eine Dimension handelt (Tabelle 3-2).

Kurzname der Variablen	Beschreibung	Datentyp	Beschreibende Dimension(en)	Einheiten
x	**1. horizontale Achse**	**float**	**x**	**Meter**
y	**2. horizontale Achse**	**float**	**y**	**Meter**
z	**Vertikale Achse**	**float**	**z**	**Meter**
time	**Modellzeit**	**int**	**time**	**Jahr**
ih	Eisdicke	float	time, y, x	Meter
rh	Topographie	float	time, y, x	Meter
eus	Eustatischer Meereshöhenwechsel	float	time	Meter
mbal	Oberflächen Massenbilanz	float	time, y, x	Meter / Jahr
pmdt	Basale Temperatur	float	time, y, x	Grad Celsius
melt	Schmelzrate	float	time, y, x	Meter / Jahr
cony	Kontinentalität	float	time, y, x	1
calv	Kalbungsrate	float	time, y, x	Meter / Jahr
slc	Meereshöhenwechsel	float	time, y, x	Meter

Tabelle 3-2 Die dreizehn Variablen in den NetCDF Beispieldatensätzen mit ihren Namen, Beschreibungen Datentypen und Dimensionen

Grösse der Dimensionen (in Rasterzellen): x = 142 ; y = 119 ; z = 11 ; time = UNLIMITED

Im Dateikörper werden alle Daten abgelegt. Die genaue Struktur der Speicherung dieser Daten braucht den Benutzer des Datenformates nicht zu kümmern. Die Datenverwaltung kann als Blackbox betrachtet werden. Die NetCDF Bibliothek liefert alle relevanten Schreib-, Lese- und Definitionszugriffe als Schnittstellen zu verschiedenen Programmiersprachen. Die Daten selber sind somit unabhängig von Plattform und Programmiersprache. Beispielsweise kann eine NetCDF Datei auf einem Linuxsystem mittels Python definiert werden. Anschliessend werden die Daten auf einem Windowsrechner mit Java geschrieben und schlussendlich auf einer Solarismaschine mit C++ ausgelesen.

3.4 Verwendung der Daten

Nicht alle in den Dateien enthaltenen Variablen lassen sich mit dem Prototypen gleich gut darstellen. Aussagekräftige Visualisierungen, die bei jedem Zeitschritt eine Veränderung zeigen, können mit den Variablen ‚Eisdicke' (ih) ‚Oberflächen Massenbilanz' (mbal) ‚Kontinentalität' (cony) und ‚Kalbungsrate' (calv) erstellt werden. Die Variable ‚Topographie' (rh) kann ebenfalls visualisiert werden, verändert sich aber in keinem der Zeitschritte, da die topographischen Daten für alle Schritte gleich belassen sind. Die Dimensionsvariablen können nicht visualisiert werden, da sie die Grenzen der anderen Variablen setzen. Ihre Darstellung ist generell nicht möglich. Da der Eustatische Meereshöhenwechsel (eus) nur von der Zeit abhängig ist, damit also keine flächige Visualisierung erlaubt, kann diese Variable nicht gezeichnet werden. Die Variablen ‚basale Temperatur' (pmdt) ‚Schmelzrate' (melt) und ‚Meereshöhenwehchsel' (slc) erzeugen keine aussagekräftige Darstellung, da ihre Werte in den Beispieldaten alle 0 sind.

4 Umsetzung

Das Ziel dieser Diplomarbeit ist nicht ausschliesslich die theoretische Betrachtung der Visualisierungsmöglichkeiten von Unsicherheitsdaten. Vielmehr soll ein weiterer Schwerpunkt bei der Entwicklung einer prototypischen Anwendung solcher Darstellungen liegen. Besonderes Augenmerk liegt in der einfachen Erfassbarkeit und der eindeutigen Interpretation der Darstellungen. Es werden einige der im Kapitel Grundlagen erdachten Visualisierungen an Beispielen praktisch umgesetzt. Anschliessend werden diese mit einer kleinen Gruppe von Testpersonen in einer hauptsächlich qualitativen Evaluation hinterfragt. Ein weiteres Ziel dieser Implementierung ist die Suche nach einer geeigneten Benutzerschnittstelle zur einfachen Generierung der Visualisierungen und einer schnellen und intuitiv gehaltenen Benutzerführung. Dabei ergeben sich einerseits Fragen zum Design der Programmoberfläche. Ihre optische Gestaltung soll von den Benutzern als visuell angenehm, verständlich und intuitiv empfunden werden. Andererseits ist die passende Auswahl der Software-container wie Listen und Knöpfe ebenso wichtig für die reibungslose Interaktion mit der Applikation.

Das erhoffte Endprodukt ist kein kommerziell vertriebenes Softwarepaket. Die Entwicklung eines solchen ist mit den gegebenen Ressourcen nicht durchführbar. Deshalb wird eine Auswahl an Funktionen und Visualisierungen zur Umsetzung bestimmt. Diese erfolgt nach subjektiv gewichteten Kriterien des Autor. Es wird versucht, so weit wie möglich auf die verschiedenen zuvor definierten Gesichtspunkte einzugehen. Die Bezeichnung als Prototypen wiederspiegelt das experimentelle Entwicklungsstadium, in der sich die Applikation befindet.

In diesem Kapitel soll die technische Seite der Umsetzung beschrieben, sowie die verschiedenen umgesetzten Visualisierungsarten erläutert werden. Die Evaluation wird anschliessend in einem eigenen Kapitel behandelt, wo ausführlich auf die unterschiedlichen Bewertungen der Testteilnehmer eingegangen wird.

4.1 Eingesetzte Technologien

4.1.1 Linux Betriebssystem

Die Applikation wurde auf einem Personal Computer mit einem Linux Betriebssystem implementiert. Die ursprüngliche Idee, ein Plugin zu QGIS, einem Open-Source GIS Projekt, zu entwickeln, gab diese Plattform vor. Nachdem dieses erste Konzept fallen gelassen wurde, der Autor sich mittlerweile in Linux eingearbeitet hatte, wurde dieses Betriebssystem nicht mehr gewechselt. Im Laufe der Umsetzung erwies sich die Plattform als sehr geeignet, da für sie viele frei verwendbare Anwendungen existieren. Deren Installation ist nicht immer einfach, und es benötigt dazu eine gewisse Einarbeitungszeit. Mit ein wenig Übung bereitet der Ablauf jedoch keine Mühe mehr. Ausserdem erfreut sich Linux als Unix Abkömmling einer grossen Beliebtheit unter Forschern. Die grosse Anzahl Software-Erzeugnisse, die frei zur Benutzung beziehungsweise zur Weiterverarbeitung im Internet zur Verfügung stehen, ist dafür mitentscheidend.

4.1.2 C++ Programmiersprache und qT Framework

Die Wahl der Programmiersprache ist wie beim Betriebssystem nicht entscheidend für das Funktionieren des Programms, hat jedoch einen Einfluss auf die Vorgehensweise und die Effizienz der Umsetzung. Deshalb war es wichtig, eine moderne, typisierte, objektorientierte Sprache zu wählen. Dies erleichtert das Arbeiten, da in klar abgrenzbaren Einheiten programmiert werden kann. Solche Einheiten werden in der objektorientierten Programmierung Klassen genannt. Jede Klasse lässt sich unabhängig von den anderen entwerfen, implementieren und kontrollieren. Alleine die Schnittstellen nach aussen müssen stimmen (Willms 1999). Dies verleit der Applikation eine Resistenz vor Abstürzen durch Fehlzuweisungen, da bereits vor dem Starten der Anwendung die korrekten Zuweisungen von Variablen kontrolliert wird. Die Syntax soll leicht verständlich und klar strukturiert sein, um zusätzlichen Aufwand durch die Anwendung einer kryptischen, schwer erlernbaren Sprache zu vermeiden.

Damit schränkt sich die Auswahl auf wenige übliche Sprachen ein, hauptsächlich Java und C++. Die Wahl fällt auf Letztere. Ausschlaggebend dafür sind die weite Verbreitung, sowie die langjährige Bewährung im täglichen Einsatz unzähliger Softwaresysteme. Die robuste Basis durch C (aus der sich C++ entwickelt hat) und die dazu natürlich gegebene Schnittstelle sind weitere Pluspunkte. Das persönliche Interesse des Autors, eine zusätzliche Programmiersprache zu erlernen, gab den letzten Ausschlag für die Wahl von C++.

Da die Kernklassen von C++ einen beschränkten Funktionsumfang bieten, ist deren Ergänzung von Vorteil. Mit qT (Trolltech 2005) kommt ein äusserst mächtiges C++ Framework zum Einsatz, das die Entwicklungsarbeit erleichtert und beschleunigt. QT stellt alle gängigen grafischen Benutzerelemente, abstrakte Containerklassen, Speicher- und Lademechanismen sowie viele weitere Hilfsklassen als Standardimplementierung bereit. Zudem bietet diese Softwarebibliothek ein ausgefeiltes Signal-Übermittlungsystem zur Kommunikation der Klassen untereinander. Dafür wird weder das Konzept der Vererbung benutzt, noch braucht eine Klasse ein Mitglied einer anderen zu sein. Dies ermöglicht eine grosse Flexibilität während der Implementierung. QT ist für Linux Freeware und darf für nicht kommerzielle Applikationen beliebig verwendet werden. Diese Bestimmung gilt ausschliesslich für die Linuxversion. Eine Portierung auf andere Betriebssysteme ist finanziell erschwert.

4.2 Die prototypische Applikation

4.2.1 Programmstruktur

Wie weiter oben dargelegt, wird für die Implementierung des Programms die objektorientierte Programmiersprache C++ verwendet. Es ist bei der Softwareentwicklung üblich, Programmstrukturen mittels UML (Unified Modelling Language) darzustellen. Die Abbildung 4-1 auf Seite 32 zeigt diese Struktur als UML-Klassendiagramm mit allen Klassen und deren wichtigsten Attributen und Methoden. Für die Logik der Applikation unwichtige Elemente wurden weggelassen oder als gruppierten Eintrag notiert.

Den Klassen-Bezeichnungen ist auf der CD das Präfix ‚UncVis' vorangestellt, um die Zugehörigkeit zum ‚Uncertainty Visualization Tool' zu verdeutlichen. Da dies alle Klassen gleichermassen betrifft, wurde in diesem Text dieser Zusatz zugunsten der Übersichtlichkeit weggelassen.

Die einzige Funktion der Main-Klasse als Einstiegspunkt zur Applikation ist die Instanzierung der Gui-Klasse. Diese erzeugt die zentrale Benutzeroberfläche mit der Menuleiste und allen zentralen grafischen Steuerelementen. Sie enthält den Vektor für sämtliche aktiven Visualisierungsfenster, instanziert die IO-Klasse und die Messages-Klasse. Die beiden Letzteren sind Hilfsklassen, welche einen eng begrenzten Funktionsumfang besitzen. Die Messages-Klasse stellt alle kleineren Informations-Fenster wie z.B. Fehlermeldungen während der Laufzeit zur Verfügung. Jede Klasse, die solche Informationsboxen zeigen soll, greift mittels Referenz auf die Messages-Instanz zu. Die IO-Klasse ist für alle Dateizugriff verantwortlich. Sie enthält die Methoden für das Einlesen der NetCDF-Dateipfade und liest die Angaben zu den Variablen und Dimensionen aus diesen Dateien. Der Fenster-Vektor nimmt als ein Kernelement der Programmarchitektur einen besonderen Platz ein. Jede neu generierte Visualisierung wird an der letzten Position dieses Containers eingefügt. Sämtliche Zugriffe auf die Visualisierungsfenster, darunter fallen alle Darstellungs-, Änderungs- und Löschzugriffe, wie auch alle Informationsabfragen betreffend der Fenster, werden mittels Iterationen über diesen Vektor gesteuert. Aufgrund differierender Datenstrukturen müssen die Fenster von Differenzenrechnungen gegenüber solchen von Originaldaten unterschiedlich angesprochen werden. Trotzdem sollen alle in einer Iteration erfasst sein. Dies wird mittels Vererbung gelöst. Der Vektor nimmt Instanzen der NetCDF-Klasse auf, die als Basisklasse der NetCDFOrig- (Originaldaten) und der NetCDFCalc-Klasse (Differenzendaten) fungiert. Die spezialisierten Unterklassen implementieren alle virtuellen Methoden ihrer Basisklasse. Somit ist es möglich, beide Klassenarten trotz deren Unterschiedlichkeit im selben Container zu speichern.

Die neusten Applikationsversionen enthalten eine weitere Klasse, die NetCDFDialog-Klasse. Sie wurde aufgrund von Reaktionen der Testpersonen während der Evaluation eingeführt und liefert einen Dialog zur Auswahl der Visualisierungsfunktionen. Neben Informationen zu Namen und Pfad der dargestellten Datei, wird die Auswahl der gewünschten Variable und das Setzen eigener Begrenzungen (Limiten) gesteuert. Auch die Farbmuster können so geändert werden.

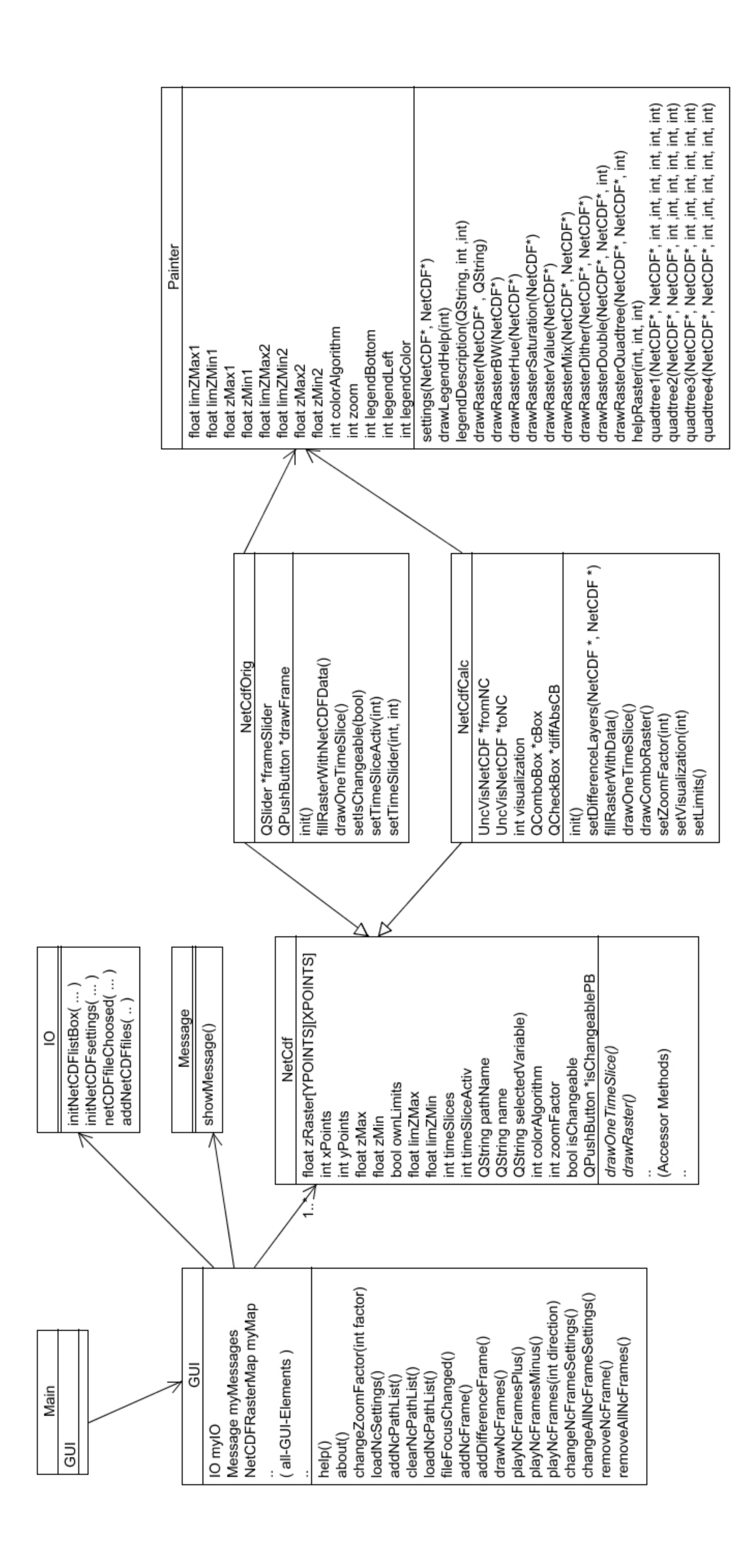
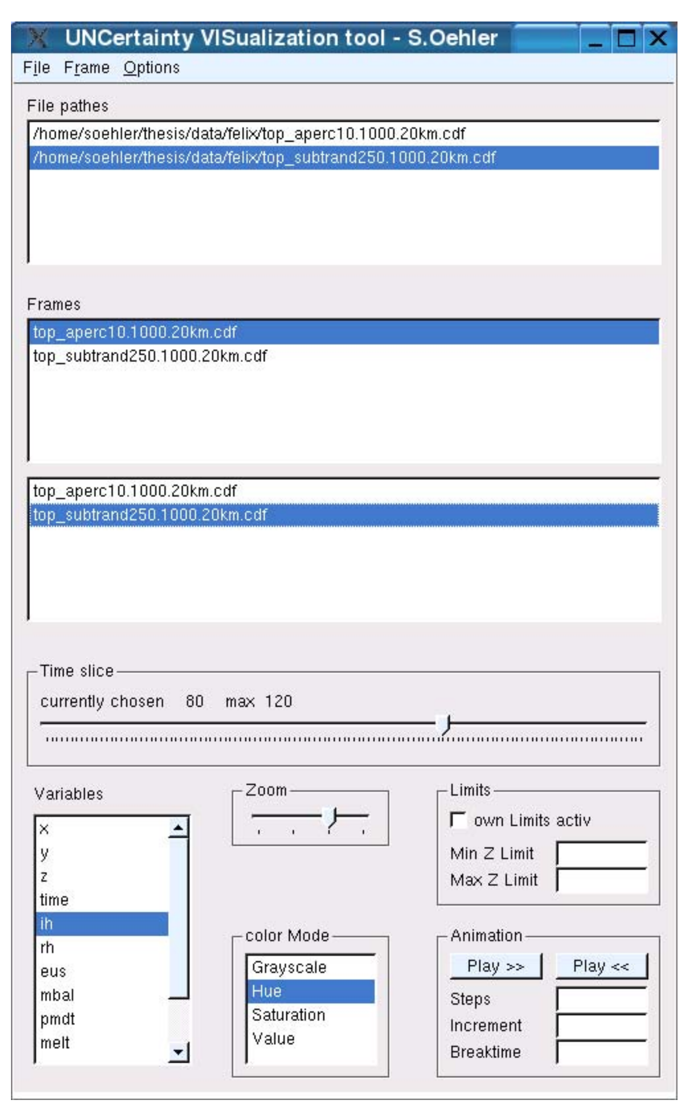


Abbildung 4-1 Die Programmstruktur als UML Klassendiagramm dargestellt. Weniger wichtige Attribute und Methoden wurden ausgeblendet.

4.2.2 Die GUI und ihre Funktionen



Die Mensch-Maschine Schnittstelle wird mit einer grafischen Benutzeroberfläche (kurz GUI für Grafical User Interface) implementiert. Erstens erscheint eine Bedienung über die Kommando-Zeile für die besprochene Aufgabe zu umständlich und kompliziert. Zweitens ist eine ansprechende GUI heutzutage ein wichtiger Bestandteil jeder Applikation. Die Umsetzung mit mächtigen Frameworks wie das hierfür benutzte qT ist, verglichen mit den logischen Kodierungsproblemen, eine Leichtigkeit. Drittens soll eine Anwendung, die zur optischen Sichtung von Daten konzipiert wird, um diese so klar und schön als möglich zu visualisieren, nicht durch ihre karge, veraltete Schnittstelle auffallen. Dies wäre ein grober Widerspruch von Form und Funktion. Cartwright et al. (2001: 46) schreiben dazu, dass zusätzlich zur Funktionalität der Oberfläche ästhetische Designaspekte berücksichtigt werden müssen, da eine ansprechende Ästhetik die anfängliche Einarbeitung fördern und die Akzeptanz des Produktes erhöhen kann.

Abbildung 4-2 Die Benutzeroberfläche (GUI) der Applikation

Der folgende kurze Überblick über die GUI beschreibt die einzelnen Elemente und die Funktionen, die damit gesteuert werden. Für eine detailierte Erklärung sei auf das Benutzerhandbuch verwiesen, das auf der beigelegten CD-Rom in deutscher und englischer Sprache vorliegt. Diese Übersicht bezieht sich auf die Programmversion 0.4.34.

Die oberste ‚File pathes' Liste enthält die Pfade zu den NetCDF Dateien, die mittels dem Menupunkt ‚Add file pathes' eingelesen werden. Mit dem Markieren einer dieser Dateien lädt die Applikation die darin enthaltene Anzahl Zeitschritte und stellt diese im ‚Time slice' Schieber dar, wo stets die Gesamtzahl aller Zeitschritte des markierten Listeneinträges zu sehen bleibt. Mit diesem Schieber lässt sich der zu visualisierende Zeitschritt direkt anwählen. Weiter werden alle in der Datei enthaltenen Variablen in die ‚Variables' Liste geschrieben. Nach der Auswahl einer Variablen muss der Programmbenutzer den gewünschten ‚Color mode' (den Farbverlauf) einstellen. Zur Zeit stehen vier verschiedene zur Verfügung: Grayscale (Grauwert), Hue (Farbton), Saturation (Farbsättigung) und Value (Farbwert). Eine genaue Erläuterung jedes einzelnen wird im Abschnitt ‚Visualisierungen' gegeben. Für die gewählte Variable lässt sich in der ‚Limits' Box optional eine Angabe eigener

Grenzwerte hinterlegen. Diese werden beim Zeichnen der Daten als minimale, respektive maximale Limite des Farbverlaufes verwendet. Werden keine Begrenzungen gesetzt, übernimmt die Applikation automatisch den Minimal- bzw. Maximalwert der selektierten Variable. Sind alle Visualisierungsoptionen gewählt, erzeugt die Applikation über den Menupunkt ‚Add new frame' eine neue Visualisierung in einem eigenen Fenster. Dessen Titel erscheint gleichzeitig in den beiden ‚Frames' Listen, die alle erzeugten Visualisierungsfenster enthalten. Diese Listen dienen zur Berechnung neuer Differenzendarstellungen. Der Benutzer muss dazu in beiden Listen ein Titel einer Visualisierung anwählen und die Berechnung mittels ‚Calculate Difference frame' aus der Menuleiste starten. Die Daten der in der unteren Liste angewählten Darstellung werden von jener in der oberen Liste subtrahiert. Durch das Umkehren dieser Auswahl wird die Differenzendarstellung invertiert.

Die GUI enthält zwei weitere Steuerungselemente. Mit dem ‚Zoom'-Schieber kann die Grösse der Rasterzellen in den Visualisierungen gesteuert werden. Die Einstellweite reicht von 1x1 bis 4x4 Pixel je Rasterzelle. Die Animationsbox erlaubt, eine frei wählbare Anzahl von Zeitschritten in kurzer Folge hintereinander darzustellen. Durch den schnellen Wechsel der Modellzustände kann der Betrachter den fliessenden Ablauf der Modellierung verfolgen.

4.2.3 Visualisierungen

Das Programm bietet zwei verschiedene Hauptgruppen von Visualisierungen. Zum einen können die Originaldaten einzeln und unverändert angezeigt werden. Dafür stehen vier verschiedene Farbgebungen zur Auswahl, drei davon sind Helligkeitsübergänge und die letzte ein Spektralfarben-Übergang. Somit kann die Anwendung als einfacher Viewer für NetCDF Dateien dienen. Zum anderen – und dies ist die spezielle Funktion, derentwegen die Applikation entwickelt wird – können zwei Datenquellen miteinander verglichen werden. Das Resultat des Vergleiches (zur Zeit eine einfache Subtraktion jeder Rasterzelle, komplexere statistische Berechnung mit mehr als zwei Quellen sind vorstellbar) kann auf mehrere Arten betrachtet werden. Einerseits kann nur die Differenz für sich alleine angeschaut werden. Diese Darstellung gleicht stark jener für die Originaldaten. Es stehen dazu die gleichen vier Farbschemen zur Verfügung. Andererseits bieten sich verschiedene kombinierte Differenzenvisualisierungen an. Dabei wird eine Originaldatei zusammen mit der Differenz dargestellt. Zwei Möglichkeiten gibt es dafür: Entweder wird die Differenz als zusätzliche Zeichnungseben über die Originaldaten gelegt, oder die beiden werden miteinander verrechnet und in einer Ebene gezeichnet. Der Begriff ‚Ebene' wird hier metaphorisch gebraucht. Die Applikation bietet nicht die von anderen Softwarepaketen bekannte Ebenenstruktur. Es sind einige Beispielvisualisierungen implementiert, die weiter unten detailliert beschrieben werden.

Gleichzeitige Darstellung in getrennten Fenstern

Da jede neue Datenansicht in einem eigenen Fenster erzeugt wird, können sämtliche Visualisierungen gleichzeitig miteinander verglichen werden. So lässt sich ein ganzes Mosaik an unterschiedlichen Ansichten der gleichen Datensätze erzeugen, um die Effekte der Farbschemen und Differenzendarstellungen zu begutachten und gegeneinander abzuwägen. Die folgende Abbildung 4-3 auf der nächsten Seite illustriert diese Situation mit acht geöffneten Visualisierungsfenstern.

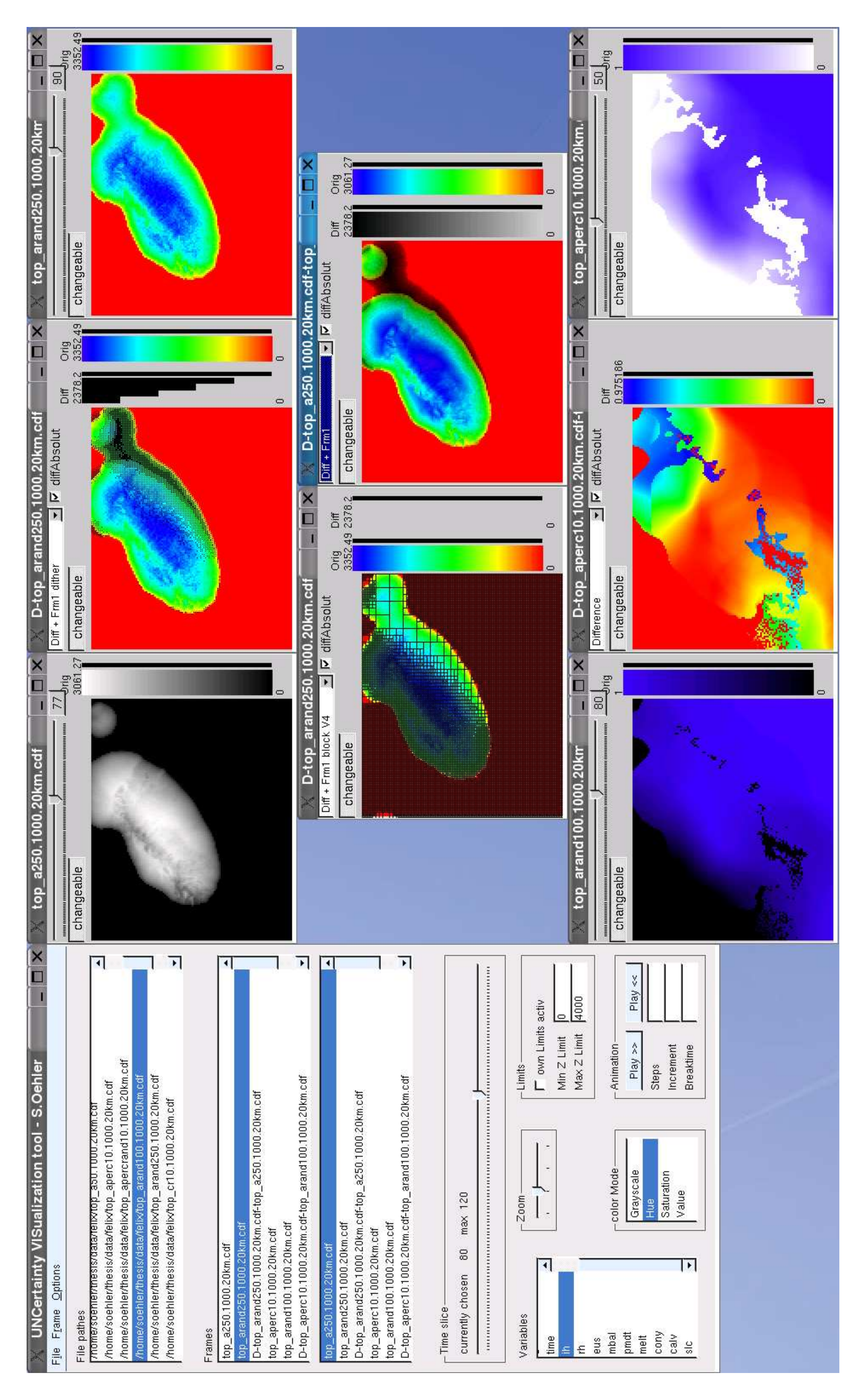


Abbildung 4-3 Die Applikation in Betrieb: Links im Bild ist die GUI zu sehen, rechts davon verschiedene Original- und Differenzenvisualisierungen.

Diese Darstellungsoption zeigt eine der besprochenen Unsicherheitsvisualisierungen, indem gleichzeitig zu den Originaldaten in einem Fenster an dessen Seite die Metadaten in einem weiteren Fenster gezeichnet werden. Sie greift das Konzept der ‚2 static maps' nebeneinander auf (van der Wel et al. 1994: 323). Diese Anwendung erweitert diese Darstellungsart dadurch, dass die Unsicherheit mehrfach gleichzeitig in anderer Form angezeigt werden kann. So können Unterschiede, die nicht in allen Visualisierungen gleich deutlich zum Vorschein treten, trotzdem erfasst werden. In der bereits verwendeten Nomenklatur kann diese Variante ‚multiple static maps and views' heissen (Abbildung 4-4).

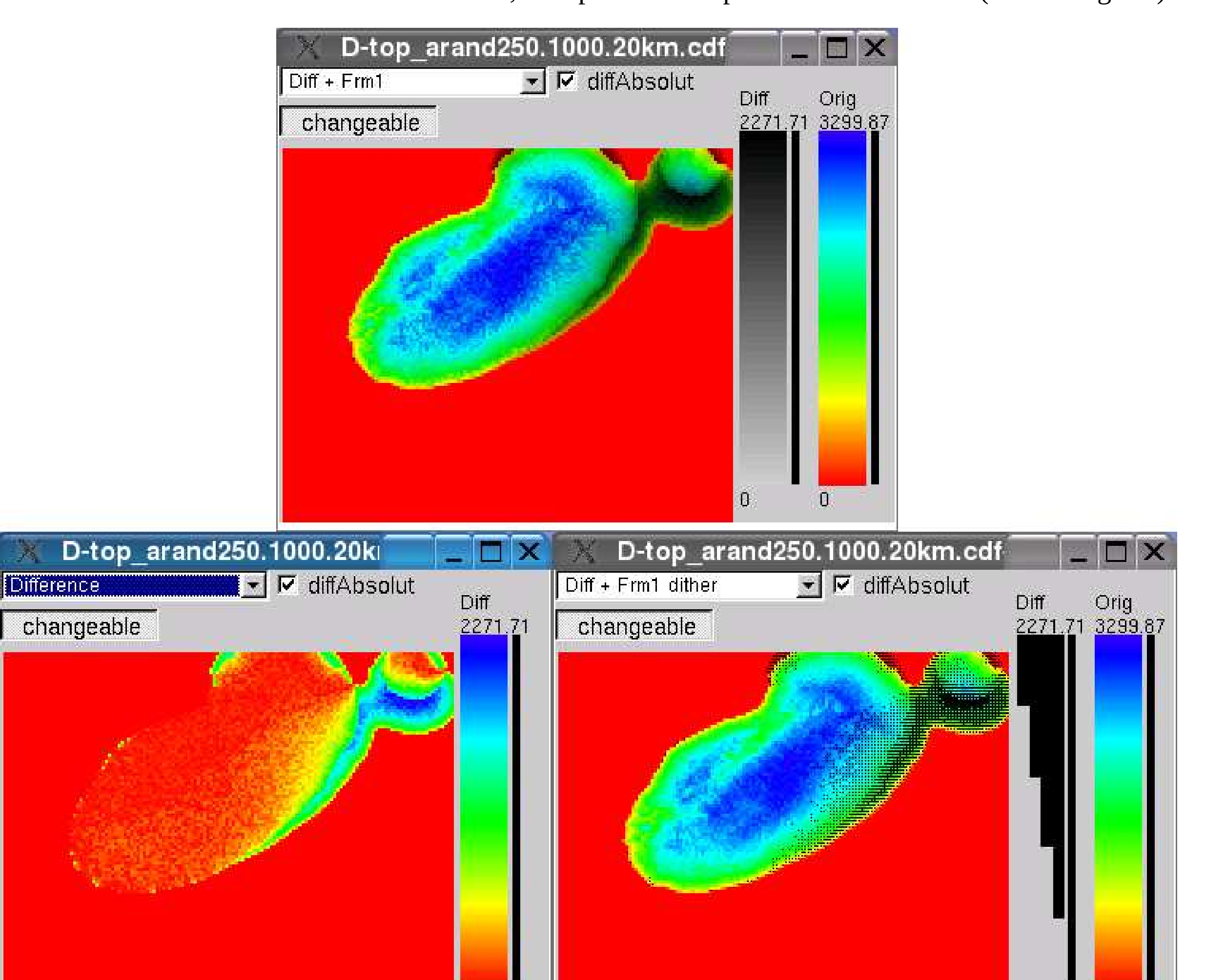


Abbildung 4-4 Drei mal die selbe Differenz: Als separate Farbton-Darstellung (links), als Dithering (rechts) und als Schattierung (oben) über die Originaldaten gelegt.

Darstellung von Originaldaten

Kleine absolute Wertunterschiede in Bereichen von kleinen Originalwerten können je nach Differenzendarstellung verloren gehen. Beim Betrachten von Unterschieden zwischen verschiedenen Modellrechnungen ist deshalb die Sicht auf die ursprünglichen Daten mit von Interesse. Zur Repräsentation dieser Daten stehen dem Programm vier einfache Farbmuster zur Verfügung, die im folgenden Bild veranschaulicht werden (Abbildung 4-5).

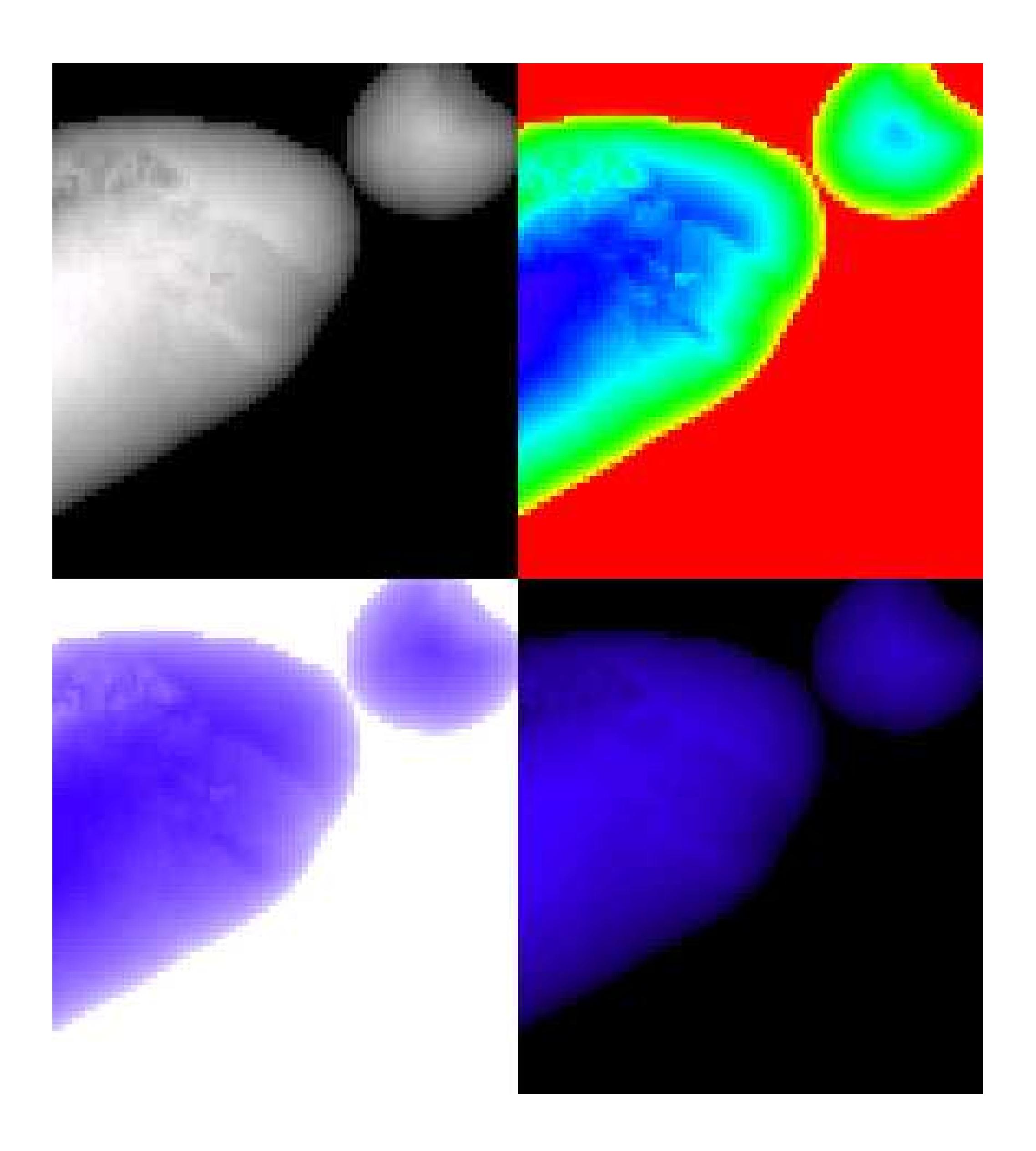


Abbildung 4-5 Originaldaten der selben Datei in den vier Farbmustern visualisiert. Graustufen (oben links) Farbton (oben rechts) Farbsättigung (unten links) und Farbwert (unten rechts)

Die Wahl dieser Farbmuster entstammt den in qT zu Verfügung stehenden Farbdarstellungsverfahren. Es gibt deren zwei, einerseits ein RGB- und anderseits ein HSV- Farbmischverfahren. Im ersten Fall entsteht die gewünschte Farbe aus dem additiven Mischen der drei Lichtfarben Rot, Grün und Blau, deren Intensität in 256 Stufen von 0 bis 100% unterteilt wird (FH-Giessen 2001: Kap 3). Die abgebildete Graustufenansicht wird mittels diesem Verfahren erzeugt. Der Wert für jede der drei Farbkomponenten wird gleich hoch gewählt, womit ein kontinuierlicher Graustufenübergang mit 256 Schritten erzielt wird. Der zweite Fall basiert auf einem anderen Konzept. Der Name HSV-Modell entsteht aus den Anfangsbuchstaben für Hue, Saturation und Value, was mit Farbton, Farbsättigung und Farbwert (oder Intensität) übersetzt werden kann (FH-Giessen 2001: Kap 4). Die Vollfarben des sichtbaren Spektrums werden auf einen Kreis aufgetragen. Pro Grad wird eine Farbe vergeben, wobei Rot auf 0° (resp. 360°), Grün auf 120° und Blau auf 240° fällt. Die Farbsättigung stellt die Blassheit des Farbtones dar. Sie liegt wiederum zwischen 0% - dies entspricht Weiss - und 100% Vollfarbe. Qt unterteilt den Parameter in 256 Stufen. Die selbe Anzahl Stufen stehen auch für den Farbwert zur

Verfügung, der die Helligkeit des Farbtones angibt. 0% entspricht vollkommenem Schwarz, während 100% wiederum die Vollfarbe repräsentiert. Farbsättigung und Farbwert beeinflussen sich gegenseitig, so dass ein Anteil von 100% Farbsättigung bei bloss 50% Farbwert keine Vollfarbe ergibt. Diese drei Parameter werden für je eine der drei restlichen Ansichten verwendet. In der Abbildung rechts oben ist die Darstellung mittels Variation des Farbtones ersichtlich. Dabei werden Farbsättigung und Farbwert bei je 100% belassen. Unten links sind Variationen mit der Farbsättigung und unten rechts mit dem Farbwert ersichtlich, wobei der jeweilig andere Parameter bei 100% belassen und der Farbton bei 120°, also reinem Blau, gewählt ist. Die Wahl des Farbtones ist nicht zwingend. Es ist ebenso gut möglich, einen beliebigen anderen zu wählen. Ohne einer Bewertung der Visualisierungen vorgreifen zu wollen, kann bemerkt werden, das gerade für die Darstellung mit dem Farbwert-Parameter ein anderer Farbton deutliche Vorteile betreffend Differenzierbarkeit der Abstufungen bringen dürfte.

Darstellung von Unsicherheiten (Differenzen)

Für die Visualisierungen der Unsicherheitsraster alleine können die selben Farbmuster wie für die Originaldaten verwendet werden. Da sich diese Darstellungen nur in der Herkunft der Daten unterscheiden, nicht aber in ihrer optischen Gestalt, wird auf eine nochmalige Erklärung verzichtet. Es handelt sich bei den Unsicherheiten in dieser Anwendung um Differenzen zwischen verschiedenen Modellen. Diese besitzen einen negativen Wert, wenn die kleineren Zellenwerte von den grösseren subtrahiert wurden. Dieser Umstand wurde mit einem speziellen ‚absolute Zahlen'-Flag berücksichtigt. Wird dieses Flag aktiviert, werden die Differenzen in absoluten Zahlen gezeichnet, also allen negativen Zahlen die Vorzeichen entfernt. In der unteren Abbildung 4-6 kann der visuelle Unterschied der Visualisierungen verglichen werden. Die gleiche Differenz ist einmal mit (links) und einmal ohne gesetztes Flag (rechts) abgebildet.

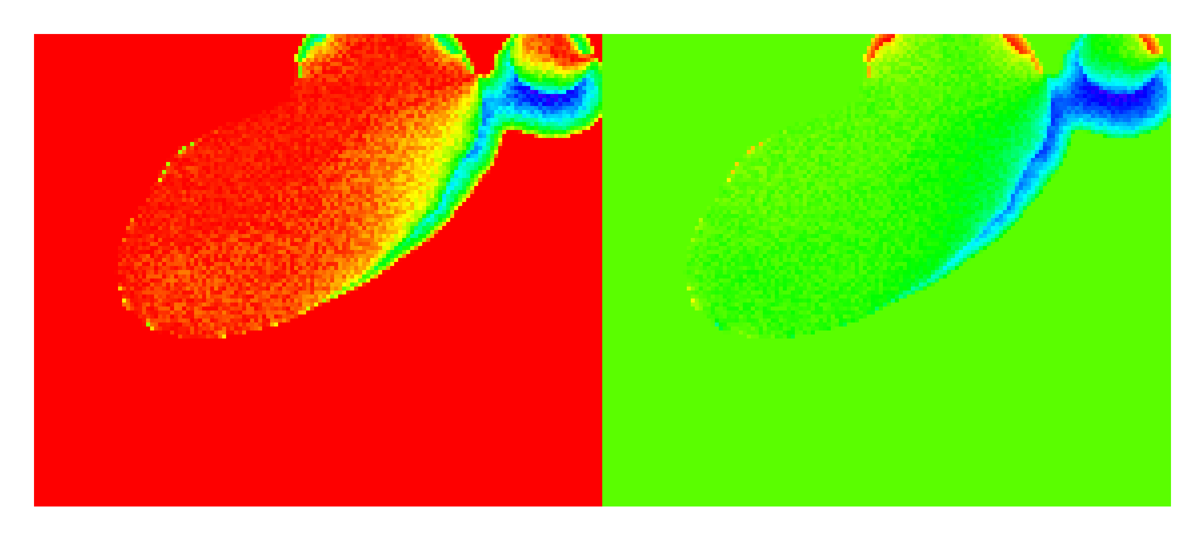


Abbildung 4-6 Zwei mal die selbe Differenz: Nur die absoluten Differenzen (links), die echten Werte reichen auch unter Null (rechts).

In der Praxis kann je nach Situation die rechte Visualisierung mit den echten Werten oder die linke Visualisierung mit den absoluten Zahlen von grösserem Interesse sein. Ist es entscheidend, in welchem Modell die kleineren oder grösseren Werte stehen, wird die rechte Darstellung mit den Vorzeichen in der Differenz zum Einsatz kommen. Will der Betrachter jedoch nur feststellen, wie die Verteilung aussieht, wählt er die linke. Besonders stark unterscheiden sich die Abbildungen, wenn die Differenzen viel tiefer in den Minusbereich als in den Plusbereich fallen.

Kombinierte Visualisierung von Original- und Unsicherheitsdaten

Ein Hauptaugenmerk der vorliegenden Arbeit liegt in der Suche und Erprobung kombinierter Visualisierungen von Originaldaten und davon abgeleiteten Metadaten. Im konkreten Fall dieser Umsetzung in der Betrachtung der stellvertretend eingesetzten Differenzendaten. Kombiniert bedeutet, das die Daten gleichzeitig in einem Fenster dargestellt werden. Dies wird mit der Bezeichnung bivariate Visualisierung ausgedrückt (van der Wel et al. 1994: 323). Es wird versucht, ein möglichst grosses Spektrum an unterschiedlichen Darstellungsvariationen zu finden. Zumindest einige davon sind in der Applikation umgesetzt. Diese werden nachfolgend vorgestellt.

Differenzendarstellung mittels Dithering über Originaldaten

In der ersten präsentierten Visualisierungsvariante wird die Dithering-Technik angewendet (Abbildung 4-7). Dabei werden je nach gewünschtem Grauton unterschiedlich viele schwarze Punkte in einer Fläche verteilt. Je mehr Punkte gezeichnet werden und je näher diese zueinander stehen, desto dunkler erscheint die Fläche. Allgemein gilt, dass bei dieser Technik mittels geschickter Punktierung versucht wird, dem Auge nicht darstellbare Farben oder Farbwerte zu suggerieren, um damit den Eindruck der gewünschten Farbe entstehen zu lassen. Einige anschauliche Beispiele dazu finden sich bei Lynch et al. (2002). Sie präsentieren in ihrem Webstyleguide diese Technik zur speicherplatz-sparenden Publizierung von Bildern auf dem Internet. Dithering wurde schon früh in der Geschichte der Computergrafik eingesetzt. Die ersten gedruckten Bilder entstanden mit Typenraddruckern, welche nur Buchstaben und einige Sonderzeichen zu Papier brachten. Es musste also ein Trick gefunden werden, die unterschiedlichen Graubereiche der Bilder als Raster zu reproduzieren. Dabei wurde ebenfalls auf die Dithering-Technik zurückgegriffen, indem nämlich an ganz dunkeln Stellen des Rasters viele verschieden Zeichen übereinander gedruckt wurden. Dazu mussten die Drucker so programmiert werden, dass diese weder nach jeder gedruckten Zeile einen Zeilenvorschub verlangten, noch die üblichen Zeilenabstände einschoben.

Die hier eingesetzte Variante des Dithering berechnet pro Rasterzelle die Modelldifferenzen. Anschliessend werden diese in einer linearen Verteilung auf die nötige Ditheringstärke je Rasterzelle umgerechnet. Je mehr Pixel pro Rasterzelle zur Verfügung stehen, desto feiner kann die Abstufung der Differenzenvisualisierung gezeichnet werden. Wird für jede Rasterzelle nur ein Pixel verwendet, kann dieser entweder in die Farbe der darunter liegenden Originaldarstellung eingefärbt werden, oder er ist schwarz. Schwarz wird er dann, wenn die Modelldifferenz an dieser Stelle grösser als die Hälfte der maximalen Differenz ist. Ein Raster mit der Zellenweite ein Pixel kann also 2 Zustände anzeigen: Originalfarbe oder Schwarz. Ein Raster mit der Zellenweite zwei Pixel kann bereits 5 Zustände einnehmen: Nur die Farbe des Originalpixels, ein, zwei, drei oder alle vier Pixel schwarz eingefärbt. Die Differenzen können also in 5 Klassen eingeteilt werden.

Kantenlänge der Rasterzelle in Pixel	Anzahl Zustände	Berechnung
1	2	1 x 1 + 1
2	5	2 x 2 + 1
3	10	3 x 3 + 1
4	17	4 x 4 + 1

Tabelle 4-1 Berechnung der möglichen Zustände einer Ditheringzelle.

Es ergibt sich eine einfache Formel für die Anzahl möglicher Zustände der Rasterzelle:

(Zellenweite in Pixel)^2 + 1

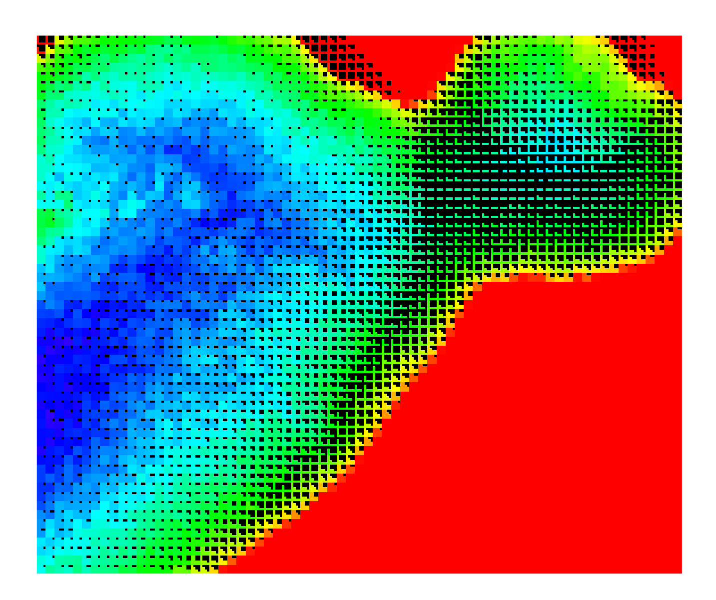


Abbildung 4-7 Die Differenz mittels Dithering über die Originaldaten gelegt.

Als dominierende Grafikvariable wird hier die Textur eingesetzt, wobei dies erst in einer aggregierten Betrachtung der Ansicht deutlich zum Vorschein tritt. Auf der Stufe des Pixels dagegen kommt der Farbwert (value) zum Einsatz: Bei einer Schwärzfärbung ist dieser 0, sonst 1 bzw. 100%. Für die Rasterzellen kann keine eindeutige Aussage gemacht werden.

Differenzendarstellung mittels Schattierung über Originaldaten

Eine mit der Dithering-Technik eng verwandte Methode ist jene der Schattierung (Abbildung 4-8). Auch hier wird der Unterschiedseffekt durch die Abdunklung der Rasterzellen erzeugt. Dazu braucht es aber keinen zweiten Layer, der über die Originaldaten gelegt wird. Der Effekt wird direkt in die Darstellung der Originaldaten gemischt. Zuerst wird die Differenz auf Zellenebene ermittelt. Nach deren Berechnung kann mittels HSV-Darstellung der Farbwert (value) als Parameter für die Stärke der Differenz verwendet werden. Ohne Setzen eigener Visualisierungsgrenzen entspricht die Maximaldifferenz aller Zellen dem Farbwert 0. Die Zelle wird ganz in Schwarz gezeichnet. Bei einer Übereinstimmung beider Modelle wird der Farbwert bei 1 belassen. Es tritt keine Abdunklung der Zelle ein. Die Abdunkelung aller anderen Zellen wird zwischen den zwei Extremwerten interpoliert.

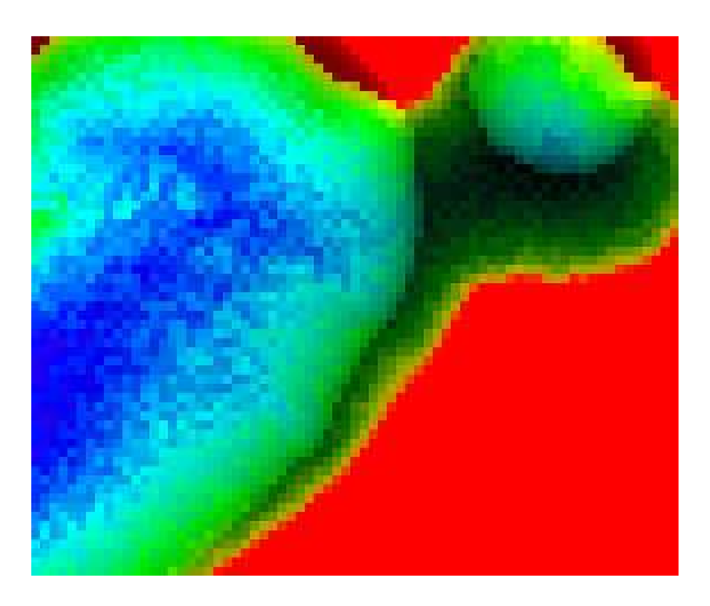


Abbildung 4-8 Die Differenz mittels Schattierung über die Originaldaten gelegt.

Eine klare Benennung der verwendeten Grafikvariable ist schwierig, weil gleich mehrere zum Einsatz kommen. Betrachtet man nur die klassischen Bertin'schen Grafikvariablen, so ist der eingesetzte Farbparameter eindeutig der Farbwert (value). Mit Einbezug der erweiterten Definition, was als Grafikvariablen gilt, muss auch das ‚Shading' (van der Wel et al. 1994) Erwähnung finden. Ein sehr ähnlicher Effekt wird bei dieser Visualisierung erzeugt. Es zeigt sich hiermit, das eine eindeutig Abgrenzung der Grafikvariablen nicht immer einfach ist.

Differenzendarstellung mittels Quadtree-Algorithmus

Gegenüber den beiden bisher vorgestellten Visualisierungen verfolgt die zu Besprechende einen etwas anderer Ansatz. Die Differenz wird dabei nicht pro Rasterzelle berechnet, sondern über eine Fläche gemittelt. Mit Hilfe eines Quadtree-Algorithmus' wird iterativ untersucht, ob die durchschnittliche Differenz unter einem definierten Schwellenwert liegt. Wenn dem so ist, wird die untersuchte Fläche geteilt, das Verfahren beginnt von Neuem. Dies wird so lange fortgesetzt, bis entweder der errechnete Durchschnitt nicht mehr unter dem Schwellenwert liegt, oder die untersuchte Fläche die Grösse von nur noch einer Zelle besitzt. Ist eine der beiden Abbruchsbedingungen erreicht, springt die Anwendung aus der Schleife und in die nächste Teilfläche des gesamten Rasters. Diese Schritte wiederholen sich, bis das ganze Raster vollständig visualisiert ist. Als Einstiegsfläche wurde ein Quadrat von 16 Rasterzellen Kantenlänge gewählt. Dieser Wert kann im Prinzip beliebig definiert werden, es empfiehlt sich aber wegen der Halbierung der Flächen ein Mehrfaches von 2. Andere Kantenlängen komplizieren den Algorithmus unnötig, da ab einer gewissen Teilungsstufe die Areale nicht mehr symmetrisch zerfallen. Mit dieser Berechnung wurden 4 Visualisierungen erzeugt, die alle eine eher ungewohnte Ansicht der Daten ergeben. Die Beschreibung erfolgt in den anschliessenden Unterkapiteln.

Die technische Umsetzung nutzt das Prinzip der Rekursion. Damit sind Programmteile benannt, die sich selber bis zum Erreichen einer genau zu definierenden Abbruchbedingung aufrufen. Bei der Implementierung einer solchen Codestruktur sollte der Programmierer vorsichtig vorgehen. Damit kann sehr leicht eine Endlosschlaufe erzeugt werden, die die gesamten Rechenressourcen eines Computers verbraucht und das System lahm legt. Oftmals ist die letzte Rettung aus einer solchen Situation der komplette Neustart des Systems.

Als Gitter über die Originaldaten gelegt

Die erste Betrachtung gilt der Variante, die die ermittelte Differenz als ein Gitter über die Originaldaten spannt. Dabei kennzeichnen grosse Maschenweiten hohe Differenzen, während ein engmaschiges Netz kleine Differenzen symbolisiert. Die unter dem Gitter liegenden Originaldaten bleiben generell sichtbar, ausser die Applikation verfügt zur Darstellung des Rasters nur über einen Pixel pro Zelle. In diesem Fall werden die Rasterzellen mit den kleinsten Abweichungen komplett vom Gitter überdeckt, da die Mindestbreite der Gitterlinien ebenfalls ein Pixel beträgt. Aus dem Netz wird in jenen Bereichen eine Fläche, die Visualisierung verliert an Aussagekraft. Bei der Nutzung dieser Visualisierung muss demnach darauf geachtet werden, die Rasterzellen mit mindestens zwei Pixel Kantenlänge zu zeichnen. Mehr Übersichtlichkeit ist allerdings erst ab einer Kantenlänge von drei Pixeln gegeben. Untenstehende Abbildung 4-9 illustriert diesen Sachverhalt. Sie zeigt drei mal den gleichen Kartenausschnitt, links mit einem Pixel je Zelle, rechts mit zwei und oben mit drei.

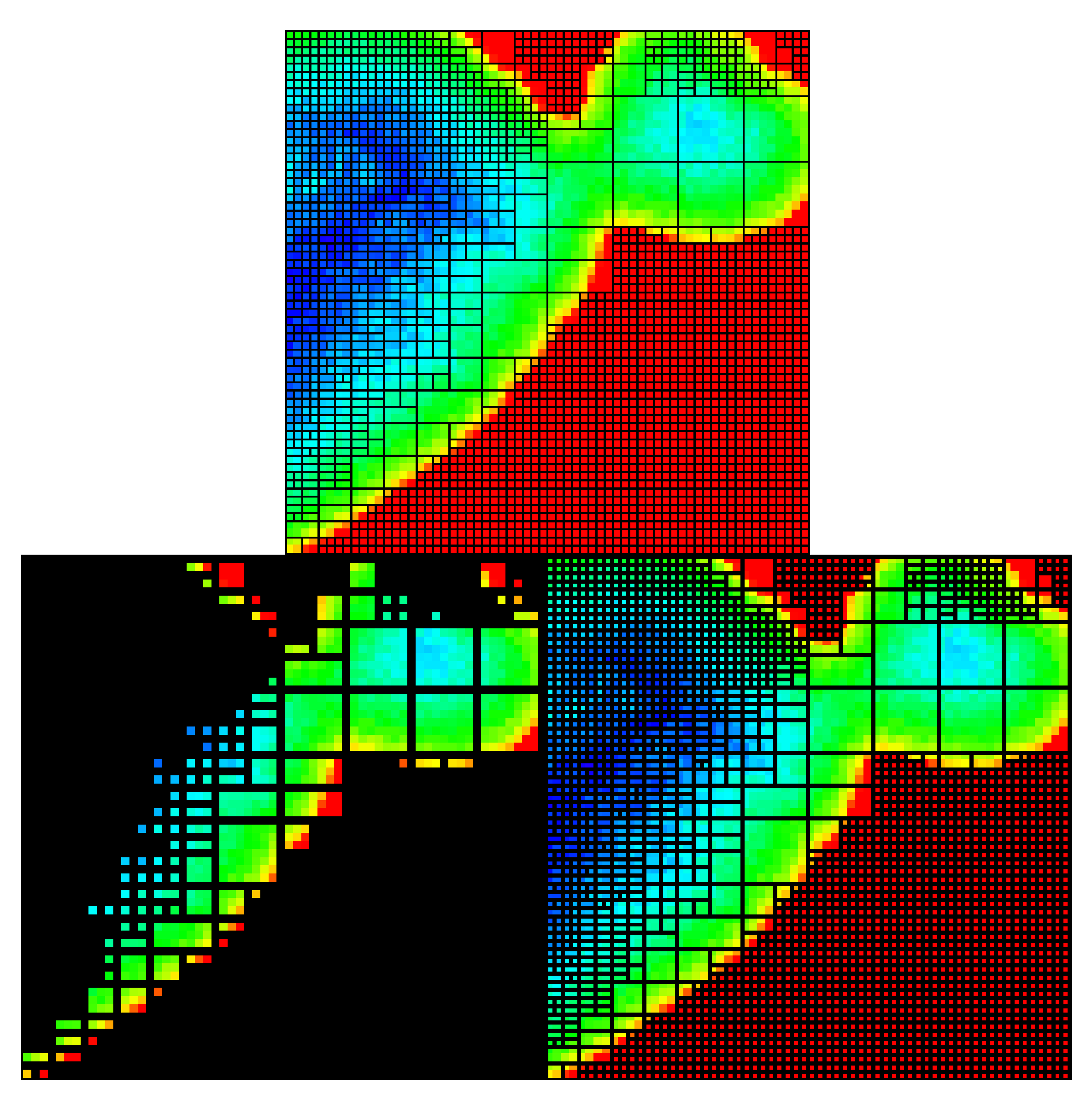


Abbildung 4-9 Vergleich dreier Zoomstufen und die Auswirkung der Liniendicke des Gitters auf die Visualisierung. Zur besseren Vergleichbarkeit wurden die Kartenausschnitte auf die gleiche Grösse skaliert.

Diese Gitterdarstellungen sind in 2 Untervarianten implementiert (Abbildung 4-10). Einerseits geschieht die Teilung mit einem echtem Quadtree-Algorithmus, das heisst die betrachteten Flächen werden nach jeder Iteration geviertelt. Daraus resultieren 5 Differenzenklassen, nämlich Gittermaschen mit 16-, 8-, 4- 2- und 1-Quadratpixeln Inhalt. Andererseits wird ein Semi-Quadtree-Algorithmus (abgewandelt nach Samet 1989) berechnet, wo die Fläche je Iteration nur halbiert wird. Dadurch entstehen fast doppelt so viele Abstufungen, nämlich 9 an der Zahl, weil nicht nur alle erwähnten Quadrate, sondern auch alle rechteckigen Zwischenstufen von 16x8, 8x4, 4x2 und 2x1 Pixeln gezogen werden. Dies ergibt eine feiner aufgeschlüsselte Visualisierung und somit einen generell höheren Aussagegehalt. Die Berechnung der Anzahl der Differenzenklassen ist einfach. Bei einem Startquadrat von 2^n Pixeln Kantenlänge, wobei n alle natürlichen Zahlen inklusive 0 sind, gilt: Anzahl Klassen bei Viertelung = n+1, Anzahl Klassen bei Halbierung = 2(n+1) – 1.

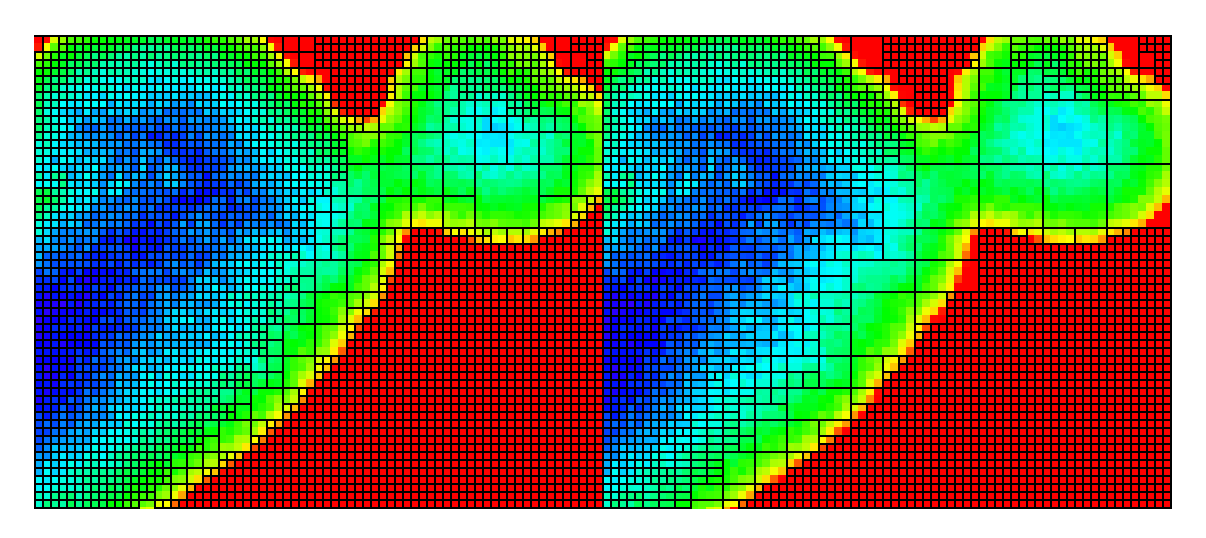


Abbildung 4-10 Die Differenz als Gitter über die Originaldaten gelegt. Vergleich der beiden Quadtree-Algorithmen: Viertelung (links), Halbierung (rechts) der Gitterzellen.

Wie schon beim Dithering findet auch hier die Textur-Grafikvariable Anwendung. Es gelten dabei die selben Bemerkungen, wie bereits weiter oben geäussert. Allenfalls ist ‚Körnigkeit' der Benennung Textur vorzuziehen. Die zwei Bezeichnungen werden oft gemeinsam oder gar synonym gebraucht.

Mit den Originaldaten vermischt

Die zweite Variante besteht aus einer Mischdarstellung, bei der eine Verrechnung der Quadtree-Aufteilung mit den Originaldaten als Basis stattfindet (Abbildung 4-11). Dabei wird der Durchschnittswert aller Rasterzellen berechnet, die unterhalb der Gitterfläche liegen, deren Ermittlung im vorhergehenden Abschnitt beschrieben ist. Sämtliche dieser Zellen erhalten beim anschliessenden Zeichnen den Farbton dieses Durchschnitts zugeordnet. Gegenüber der Gitterdarstellung besteht der Vorteil, dass auch bei einer Rasterzellenweite von einem Pixel noch alle Zellen zu sehen sind. Dagegen muss der Nachteil von teilweise grossen, einheitlichen Gebieten, in denen ein Informationsverlust auftritt, in Kauf genommen werden. Dieser Informationsverlust variiert in Abhängigkeit von der Heterogenität des zusammengezogenen Datenfeldes. Wie schon bei den Gitterdarstellungen werden der Echte- und der Semi-Quadtree-Algorithmen eingesetzt. Es gelten die gleichen Bemerkungen zu den Differenzenklassen.

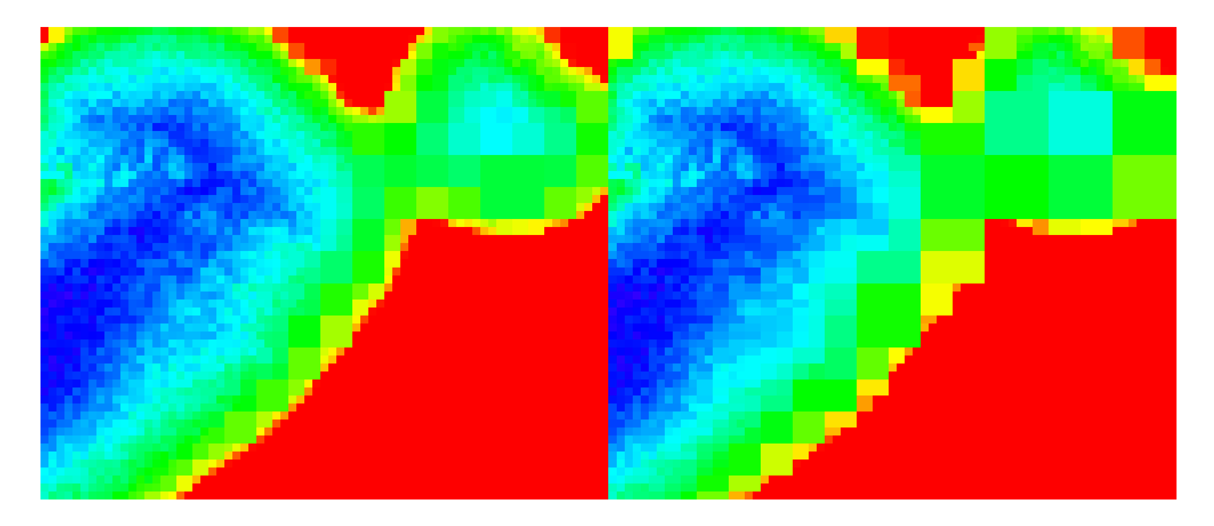


Abbildung 4-11 Die Differenz als Blockdarstellung mit den Originaldaten verrechnet. Vergleich der beiden Quadtree-Algorithmen: Viertelung (links), Halbierung (rechts) der Blöcke.

Die in dieser Visualisierung verwendete Grafikvariable trägt die Bezeichnung ‚Level of Abstraction', übersetzbar mit Abstrahierungsgrad. Durch den Zusammenzug einzelner Zellen zu ganzen Rechtecksflächen entstehen Gebiete mit grösserem Abstraktionsgehalt. Selbst die optische Erscheinung der Darstellung unterstreicht diese Einschätzung.

Darstellung zweier Originaldatensätze gleichzeitig

Es lässt sich streiten, ob diese Visualisierungsmöglichkeit die Bezeichnung Differenzendarstellung verdient. Anstatt die Differenz als solche zu visualisieren, lagern hier die Originaldaten zweier Modellierungen übereinander (Abbildung 4-12). Ein Datensatz wird in einer reinen Blau-, der andere in einer reinen Rot-Abstufung dargestellt. An den Stellen, wo sich die beiden Modelle mit hohen Werten überdecken, entsteht durch die Farbmischung ein Hellrosa Feld. Bei tiefen Werten, oder in Bereichen mit Daten nur eines Modells, ist die Visualisierung eher dunkel. Mit ihr lässt sich relativ rasch ein Überblick gewinnen, wie zwei verschiedene Raster betreffend ihrer Deckungsfläche zueinander stehen. Eine quantitative Aussage dagegen kann und soll nicht gemacht werden. Es ist zu beobachten, dass die Darstellung bei homogenen Rastern eine höhere Aussagekraft besitzt, als bei Rastern mit starken Werteschwankungen. Die Visualisierung wird von Vorteil für den Vergleich zweier Modelle mit deutlichen Unterschieden angewendet, wo sie eindeutig bessere Dienste zu leisten vermag, als bei sich stark gleichenden Modellen.

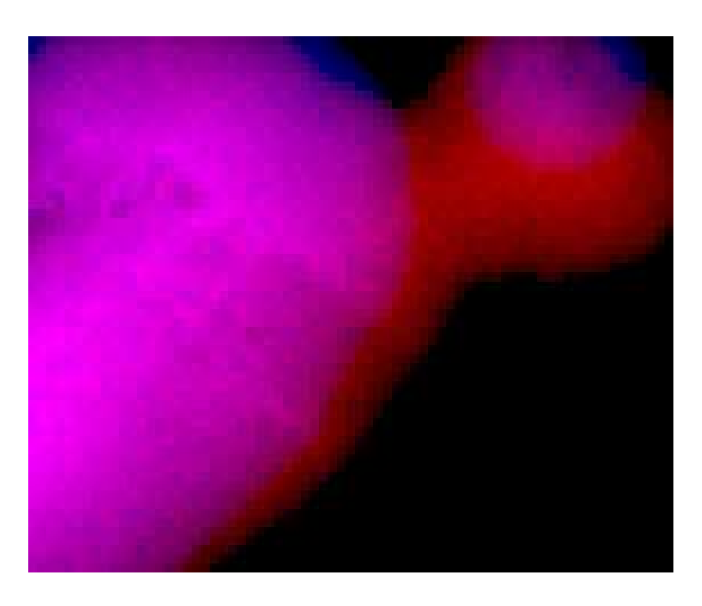


Abbildung 4-12 Zwei Originaldatensätze in einer RGB-Visualisierung gemischt: Erster Datensatz wird in Rot, zweiter in Blau gezeichnet.

Denkbar ist auch eine Variante mit drei gleichzeitig visualisierten Modellen, wobei das Dritte die Grün-Komponente des RGB-Spektrums bezieht. Diese Möglichkeit findet jedoch in der entwickelten Applikation keine Umsetzung. Dies hauptsächlich deshalb, weil dabei keine grundlegend andere Ansicht der Daten als bei der Variante mit 2 Modellen möglich ist. Zudem erscheint der Gehalt einer solchen Visualisierung eher fragwürdig, weshalb auf den zusätzlichen Entwicklungsaufwand verzichtet wurde. Interessanter wäre ein Vergleich der anderen Farbkombinationen für zwei Modelle, also Rot-Grün und Blau-Grün.

Die verwendeten Grafikvariablen sind noch einmal der Farbton (hue), um die beiden Datensätze untereinander zu trennen, und der Farbwert (value) für die Intensität der Werte beider Modelle. Bemerkenswert, dass diese so klar zu erkennen ist, obwohl die Implementierung das RGB-Schema gebraucht.

5 Evaluation

Mit der Implementierung einer Applikation, die Differenzendarstellungen herstellen kann, ist eine der beschriebenen Zielsetzungen erreicht. Als nächster Schritt folgt die Abklärung, ob das Programm und die generierbaren Visualisierungen tatsächlich den angestrebten Nutzen bieten. Dieser besteht in der Lieferung neuer Dateneinsichten und der damit verbundenen Generierung eines Mehrwertes. Die im Kapitel beschriebene Evaluation hat ausserdem zum Ziel, die Gestaltung und die Funktionalität der Applikation zu überprüfen.

5.1 Beschreibung des Testverfahrens

Es zeigt sich, dass eine tiefgreifende Prüfung der Software im Rahmen dieser Diplomarbeit nur mit einem unverhältnismässig grossen Aufwand zu erzielen ist. So beschreibt Ince (1995) in seinem Buch einen sehr detaillierten Entwicklungsprozess mit vielen Kontroll- und Testmechanismen. Die Anwendung dieser Standardprozeduren eignet sich jedoch besser bei grösseren Programmentwicklungen mit vielen involvierten Akteuren, bei denen genaue Spezifikationen und Definitionen über Strukturen und Prozesse für das Gelingen des Projektes elementar sind. Bei der hier vorliegenden Applikation handelt es sich dagegen um einen Eine-Person-Prototypen, quasi eine Vorstudie zu einem späteren grossen Softwareprojekt. Für diesen Prototypen ist die Durchführung der von Ince beschriebenen Projektplanung nicht zweckmässig.

Frühauf (2004) beschreibt eine andere Möglichkeit zur Überprüfung von Software, nämlich sogenannte Software-Reviews, von denen es ein Vielzahl verschiedener Arten gibt. Die Auswahl reicht von kleinen, informellen Zusammenkünften weniger Personen bis zu gross angelegten, mehrstufigen Prüfsitzungen, in deren iterativem Prozess verschiedene Teams zu unterschiedlichen Zeitpunkten verschiedene Rollen einnehmen. Es gilt, je grösser der betriebene Aufwand ist, desto grösser wird der Ertrag. Das heisst, je mehr Personen in die Überprüfung involviert sind, desto mehr Fehler werden entdeckt und ein um so höheres Verbesserungspotential resultiert daraus. Der Vorteil solcher Reviews besteht darin, dass bei einem geschickt geplanten Aufbau mit relativ wenig Aufwand achtbare Resultate erzielbar sind. Der Autor entschied sich für die Anwendung eines adaptierten ‚Structured Walktrough' ohne Vorbereitung der Testpersonen (Frühauf 2004:83/84). Dabei wird das Testobjekt in einem Schritt-für-Schritt Verfahren begutachtet, wobei dessen Urheber als Testleiter fungiert. Das Verfahren wurde nicht wie von Frühauf beschrieben auf die Logik und den Code des Programms angewendet, sondern auf das Programm selber, bzw. dessen Funktionalitäten. Im Fokus des Testes liegt der Benutzer und seine Wahrnehmung der Applikation, die technischen Aspekte werden nur am Rande gestreift, abhängig vom Vorwissen der jeweiligen Testperson. Jeder Test ist mit jeweils nur einer Person bestückt. Diese kann so am eingehendsten befragt und damit ein Maximum an Information gewonnen werden.

5.1.1 Aufbau der Befragung

Die Befragung gliedert sich in einzelne Teile, die aufeinander Bezug nehmen. Da neben den Visualisierungen auch ein Test der Applikation sowie der Dokumentation stattfinden soll, baut die

Testplanung darauf auf. Als Erstes sollen die Gutachter das Manual in der Weise gleichen studieren, wie sie das üblicherweise bei anderen Anwendungen tun. Sie werden ausdrücklich darauf hingewiesen, nicht gründlicher vorzugehen als bei anderen, neu zu lernenden Softwareumgebungen. Anschliessend findet der Applikationstest statt, indem untersucht wird, welche Funktionen die Tester nun beherrschen und bei welchen sie trotz dem Studium der Anleitung Mühe bekunden. Folgende Aufgaben werden gestellt:

1.) Einige NetCDF-Dateipfade in die ‚pathes'-Liste eintragen. Die Tester erhalten den physischen Pfad zu den Beispieldateien vom Testleiter.
2.) Erstellen einer Visualisierung mit Originaldaten.
3.) Wechsel der dargestellten Variablen und des Farbverlaufes in der unter Punkt 2 hergestellten Visualisierung.
4.) Verschieden Zeitschritte anwählen und darstellen.
5.) Erstellen einer weiteren Visualisierung mit Originaldaten, um eine Differenzendarstellung mit der Visualisierung von Punkt 2 zu realisieren.
6.) Erprobung unterschiedliche Differenzen-Visualisierungen.
7.) Entfernen eines Visualisierungsfensters aus der ‚Frames' Liste.

Treffen die Testperson auf ein Problem, das den Testablauf unterbricht, hilft ihnen der Testleiter mit Tipps weiter. Dies soll so erfolgen, dass die Testenden möglichst viel selbst ausführen müssen. Nach dem Erfüllen aller Aufgaben stellt der Testleiter jene Funktionen und Visualisierungen vor, die nicht zum Testprogramm zählen. Darunter fallen die Animationsfunktion und das Setzen von eigenen Grenzwerten. Je nach Interesse und einhalten des Zeitplanes können diese Funktionen durch die Gutachter ausprobiert werden. Nachdem diese das gesamte Programm kennen, erhalten sie einen kurzen, standardisierten Fragebogen (siehe Ahnhang), den sie ausfüllen. Die Auswertung dazu erfolgt jeweils am Anfang der Abschnitte des nachfolgenden Unterkapitels 4.2. Resultate. Den Abschluss der Evaluation bildet ein offenes Interview, indem folgende Punkte noch einmal spezifisch angesprochen werden:

1.) **Dokumentation:** Das Hauptaugenmerk liegt bei zwei Punkten. Erstens soll die Anleitung gut strukturiert und übersichtlich sein. Unnötige Textstellen sollen eliminiert und unklare Formulierungen vereinfacht werden. Zweitens müssen alle wichtigen Elemente und Funktionen so beschrieben sein, dass ein Benutzer das Programm alleine erlernen kann. Fehlende Informationen werden später in der Anleitung ergänzt. Ein weiteres Diskussionsthemen ist das Design der Anleitung, welches die Einfachheit und Klarheit derselben unterstützen muss.
2.) **GUI / Oberfläche:** Zentral für die Steuerung der Applikation sind die Eindeutigkeit der verwendeten Elemente, die klare Struktur der Oberfläche sowie deren leichte Erlernbarkeit. Die Bedienung soll möglichst einfach und intuitiv funktionieren. Die Funktionen sollen mit wenigen, logischen Klicks erreichbar und die benötigten Anzeigen passend platziert sein.
3.) **Funktionalität:** Es findet eine Befragung zur Nützlichkeit der vorhandenen Funktionen statt. Zum Beispiel wird abgeklärt, ob diese die von den Benutzern eingesetzt werden und die erwarteten Resultate liefern. Ausserdem können die Testpersonen ihrer Phantasie freien Lauf lassen und Ideen zu Funktionalität äussern, die sie sich zusätzliche wünschen.
4.) **Visualisierungen:** Aufgrund der Wichtigkeit dieses Themenkomplexes ist er unterteilt. Jede Visualisierung wird einzeln diskutiert und nach deren Verbesserungspotential geforscht. Der Gesamteindruck aller Visualisierungen, deren Anzahl und Praxisnähe bildet die nächste

Beurteilungsebene. Zum Schluss werden Ideen für weitere Visualisierungen und Farbverläufe aufgenommen. Wiederum sollen die Testenden kreativ tätig sein und auch unrealisierbare Wünsche anbringen dürfen.

5.) **Weiteres:** Dieser Punkt gibt den Testpersonen die Möglichkeit, sich zu nicht angesprochene Themen äussern zu können. Technische Aspekte können analysiert oder über den konzeptioneller Hintergrund der Arbeit philosophiert werden.

Dieser Testablauf ist idealtypisch und die Einhaltung der Reihenfolge nicht immer durchführbar. Dieser Umstand ist jedoch vernachlässigbar, da in dieser Evaluation vor allem die qualitativen Bemerkungen der Tester interessieren, um daraus Anregungen für die Verbesserung der erwähnten Punkte zu gewinnen. Der Testleiter beharrt nicht auf der genauen Einhaltung der Befragungsabfolge. Manche Testpersonen probieren bereits während dem Lesen der Anleitung das Gelesene an der Anwendung aus. Dieses Verhalten wird unterstützt, da es der Alltagssituation eines Anwenders entspricht, die ohnehin nur schwer simulierbar ist.

5.1.2 Testgruppe

Die entwickelte Anwendung spricht ein Fachpublikum mit geografischem Hintergrund an, das sich mit der Darstellung räumlicher Daten beschäftigt. Deshalb wurden die Testpersonen im näheren universitären Umfeld gesucht. Die GIS Abteilung des Geografischen Institutes Zürich führt einmal jedes Semester einen ‚Round Table' durch, um einen Austausch über die laufenden Forschungsarbeiten zu ermöglich. Bei dieser Gelegenheit gelangte der Autor mit einem Aufruf zur Beteiligung an der Evaluation an die Anwesenden. Es meldeten sich 8 Interessierte. Alle Evaluationsteilnehmer verfügen somit über zumindest grundlegende, einige auch über vertiefte Kenntnisse betreffend Geografischer Informations-Systeme. Geschultes Kartografisches Wissen brachten mindestens zwei der Testpersonen mit. Die Aufteilung nach Geschlechtern ist nicht ganz ausgewogen, spielt in diesem Zusammenhang aber eine untergeordnete Rolle. Bei der Evaluation nahmen 2 Frauen und 6 Männer teil. Die Befragung erfolgte in zwei Sprachen: Die Mehrzahl der Teilnehmer gehören der deutschen Sprachgruppe an, 2 Personen sprechen Englisch. Für Letztere wurde der Standardfragebogen sowie das Manual übersetzt.

Die Gruppengrösse mag eher klein erscheinen, allerdings gelte sie, laut einem Gespräch mit dem Diplomarbeitsbetreuer, in der Literatur bei dieser Art der Durchführung als sinnvoll. Bei mehr Befragungen sei kaum noch mit weiteren differenzierenden Aussagen zu rechnen. Dies bestätigte sich bei der vorgenommenen Evaluationen. Zudem ist das angewendete Verfahren zeitintensiv, ein Umstand, der ebenfalls gegen eine Erhöhung der Testanzahl spricht.

5.2 Resultate

Um eine standardisierte Vergleichsoption für die Applikation zu ermitteln, erhalten die Testpersonen einen Fragenbogen, den sie vor der Diskussion ausfüllen. Die Fragen behandeln generelle Aspekte zu den 4 Themen Dokumentation, GUI, Funktionalität und Visualisierungen, die in je einem eigenen Unterkapitel vorgestellt werden. Diese schriftliche Befragung dient einer allgemeinen Standortbestimmung zu einigen ausgewählten Punkten, die einen generalisierten Gesamteindruck ergeben. Der Fragebogen ist im Hinblick auf die kleine Stichprobengrösse bewusst einfach gehalten.

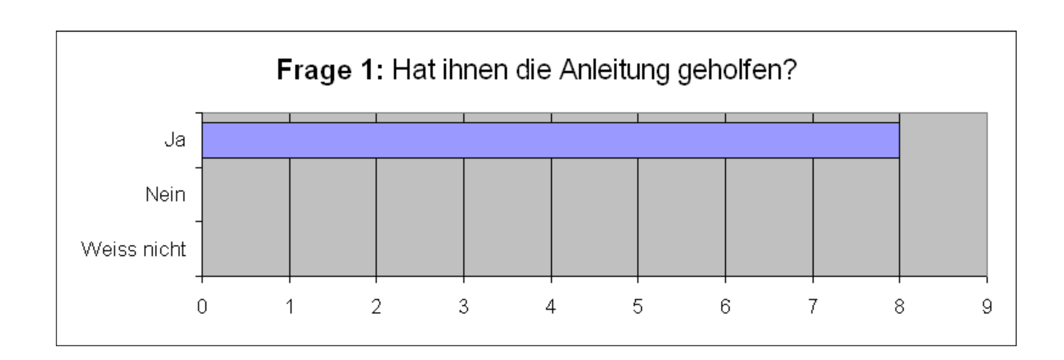
So gibt es pro Frage bloss drei Antwortmöglichkeiten wie ‚Ja', ‚Nein' und ‚Weiss nicht'. Auf den Einsatz einer feinere Werteskala zur statistischen Auswertung wird verzichtet. Das kreierte Frageformular ist im Anhang X beigefügt.

Im Anschluss an die tabellarische Auflistung der schriftlichen Fragen erfolgt die Präsentation der in der mündlichen Befragung gesammelten Aussagen. Das Ziel ist, ein breite Auswahl an Verbesserungsvorschlägen und -wünschen auszubreiten, um die vielen noch offenen Möglichkeiten vorzustellen. Andererseits sollen die als gut erachteten Punkte aufgelistet und für etwaige Nachfolgeuntersuchungen nutzbar gemacht werden. Die bereits vorgestellten 4 Themenbereiche bilden gleichermassen den Leitfaden, so dass eine themenspezifische Gruppierung der zwei Befragungseinheiten erfolgt. Bemerkungen der Testenden, die vor dem Gespräch gefallen sind, zum Beispiel während dem Ausführen der gestellten Aufgaben oder dem Lesen der Anleitung, fanden ebenfalls Eingang in die selbe Resultat-Präsentation.

Die individuelle Befragung bietet den Vorteil, das genauer auf die Aussagen jeder Person eingegangen werden kann. Durch ein Nachhaken bei jenen Punkten, die dem Tester besonders auffallen, sind sein individuelle Erfahrungsschatz und sein Wissen besser integrierbar. So lässt sich eine Thematik vertiefen, die sich als besonders ergiebig erweist. Andere wiederum können weggelassen werden, wenn der Gesprächspartner sich einer Äusserung enthalten möchte.

5.2.1 Dokumentation

Aus eigener Erfahrung ist bekannt, dass vor der Erprobung einer neuen Applikation nicht gerne viel gelesen wird. Enthält das Manual viel Text, erscheint das Lesen desselben als mühsam. Dagegen spricht das sofortige Ausprobieren der Anwendung den Spieltrieb an. Dieses Verhaltensmuster kann bei den meisten Testpersonen beobachtet werden. Nur wenige studieren die Anleitung vertieft und nehmen sich ausreichend Zeit dafür. Die Mehrheit probiert nach kurzem Überfliegen direkt die Anwendung aus. Eine kernige, knapp gehaltene Dokumentation stand deshalb bei deren Konzipierung im Zentrum. Trotzdem sollen alle Funktionen und Programmoptionen umfassend genug erklärt werden. Es ist oftmals schwierig abschätzbar, wie viel Text bereits zu viel ist, den Lesenden also durch den blossen Anblick abschreckt. Eine Alternative und Auflockerung dazu bietet der Einsatz von Bildern. In diesem Manual werden Bildschirmfotos der GUI und aller herstellbaren Visualisierungen mit einer Erklärung der abgebildeten Elemente verwendet.



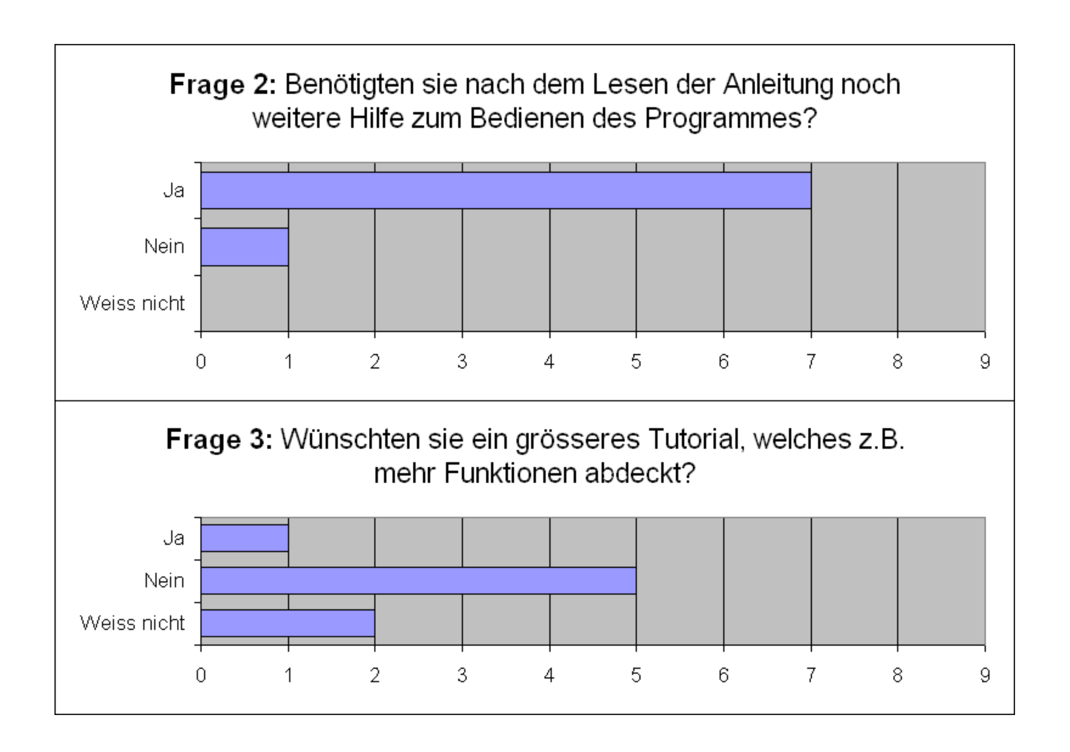


Tabelle 5-1 Auswertung der schriftlichen Fragen 1-3 des Fragebogens

Die obenstehende Tabelle 5-1 zeigt die Fragen Nummer 1 bis 3 des Fragebogens. Sie behandeln die Dokumentation und sollen Auskunft darüber geben, ob diese in der jetzigen Form Erstnutzern erlaubt, alleine einen Einstieg ins Programm zu finden. Frage 1 lotet einen bereits allfällig vorhandenen Nutzen aus, während Frage 2 darauf abzielt, ein Verbesserungspotential zu ergründen. Die Ja-Antworten dominieren beide Male. Die Anleitung unterstützt demnach den Einstieg. Zur Bedienung der Applikation braucht es aber noch mehr Information. In Frage 3 interessiert speziell das Kapitel ‚Tutorial', das wie in anderen Manuals üblich, die ersten Schritte erleichtern soll. Die Länge genügt, da sich die Mehrheit für ein Belassen in der jetzigen Grösse ausspricht.

Nicht alle Bemerkungen sind für eine spätere Anpassung des Dokumentes von gleicher Wichtigkeit. Gewisse Hinweise auf fehlende Informationen sind falsch, da diese teilweise vorhanden sind, aber beim überfliegenden Lesen des Dokumentes überlesen wurden. Die Rückmeldungen aus der Befragung gliedern sich in zwei Teile. Zum einen sind das formale Aspekte, die im anschliessenden Absatz folgen, zum anderen inhaltliche.

Die Anleitung liegt als Hypertext-Dokument vor. Ihre Formatierung verwendet Standard html-Tags, die jeder moderne Browser anzeigen kann. Das Design findet allgemein guten Anklang. Es wird als optisch ansprechend eingestuft, die gewählten Farben als übersichtlich und gut strukturierend gelobt. Dagegen empfinden nicht alle Testpersonen die Reihenfolge der Kapitel als logisch: Eine Mehrheit möchte das Tutorial erst nach dem Übersichtskapitel, das die Bedienelemente beschreibt, einfügen. Die Navigationselemente (Verlinkung innerhalb des Dokumentes) werden eher spärlich eingesetzt, so dass deren Erweiterung vereinzelt empfohlen wird.

Inhaltlich werden folgende Punkte besonders oft angesprochen. Das Kapitel ‚Vorbemerkungen' erachten die meisten Testpersonen als zu schwierig und umständlich. Sie relativieren die Aussage aber damit, dass ihnen das nötige Hintergrundwissen zu den angesprochenen Themen Unsicherheitsvisualisierung und NetCDF Datenformat fehlen. Zur Abhilfe soll entweder das Kapitel

ergänzt, oder gekürzt werden. Die Verbesserungsvorschläge fallen hier stark auseinander. Es wird bemerkt, dass spätere Benutzer einen besseren Zugang zum verwendeten Datenformat haben und dies deshalb nicht so ausführlich erklärt werden muss. Das Kapitel ‚Tutorial' soll besser strukturiert werden. Eine separate Erklärung jeder Funktion des Programms wird der jetzigen Form eines mehr oder weniger zusammenhängenden Textes vorgezogen. Das entspräche mehr der gewohnten Form eines Tutorials. Für jede einzelne Funktion soll eine klar abgegrenzte Schritt-für-Schritt Anleitungen bestehen. Der Umfang soll jedoch nicht wachsen, was bei den Vorschlägen schwierig zu halten ist. Besonders positiv erwähnt werden die „knappe" und „knackige" Art des Kapitels ‚Benutzerinterface', das mit wenigen Worten viel Information zu vermitteln mag, sowie das Kapitel ‚Visualisierungsbeispiele'. Bei Letzterem dürften die Beschreibungen zu den Beispielen ausführlicher sein. Es soll erklärt werden, was genau auf den Abbildungen zu sehen ist und wie die Visualisierungen zu Stande kommen. Ausserdem soll der angewählte Listeneintrag genannt sein, mit dem die Darstellungen verknüpft sind.

5.2.2 GUI

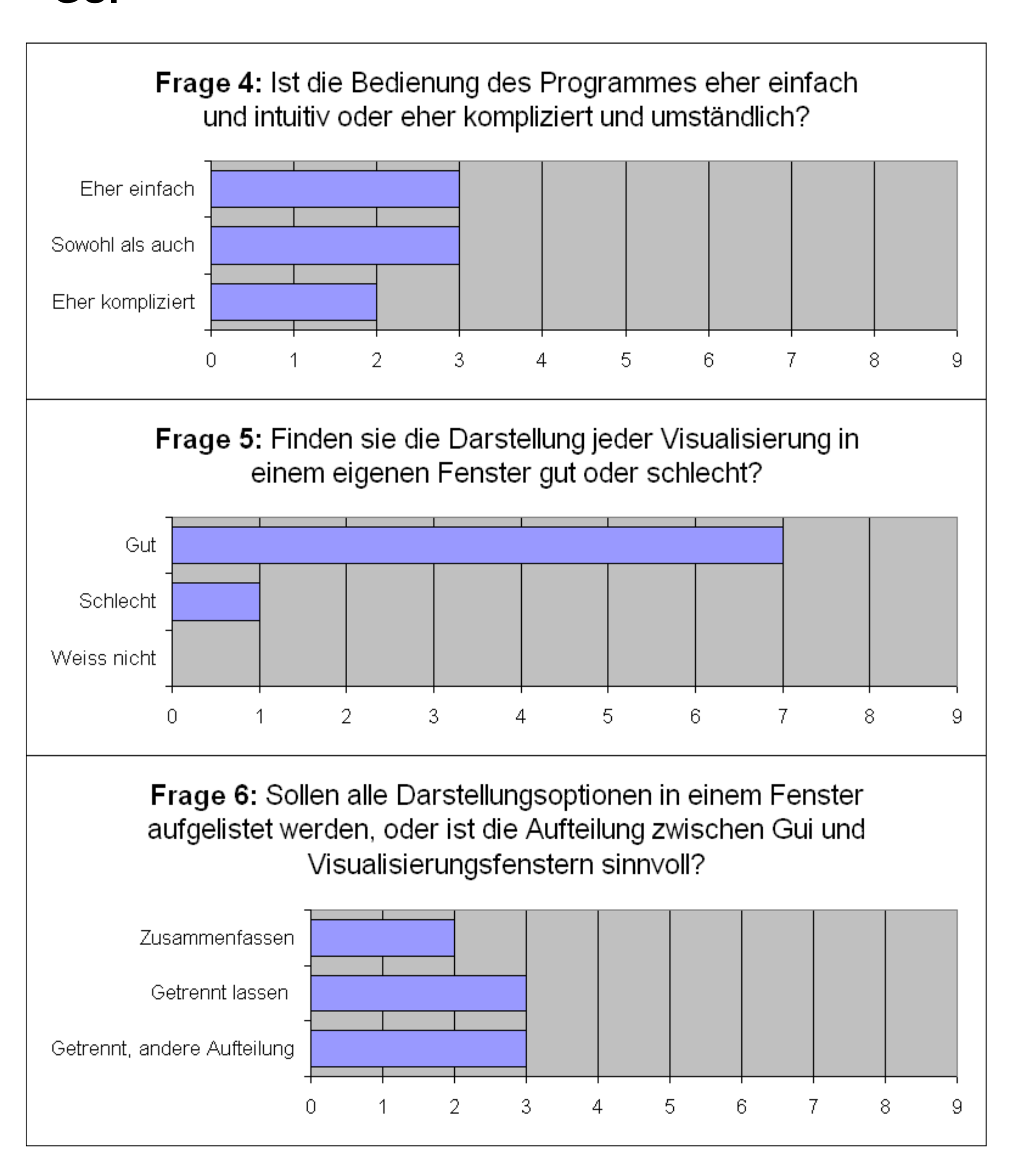
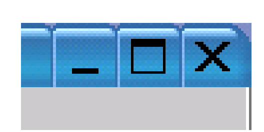


Tabelle 5-2 Auswertung der schriftlichen Fragen 4-6 des Fragebogens

Die nächsten 3 schriftlichen Fragen, abgebildet in der Tabelle 5-2 oben, betreffen Aspekte der Benutzeroberfläche und der Interaktionsmöglichkeiten. Frage 4 und 6 untersuchen die Eignung der implementierten Programm-Steuerung. Frage 4 richtet den Blick auf die GUI als Ganzes, während Frage 6 sich auf die Trennung der Visualisierungsoptionen in Hauptfenster und Visualisierungsfenster fokussiert. Die gleichmässige Verteilung der Antworten zeigt an, dass hier noch einiges verändert werden muss. Rund zwei Drittel der Testpersonen empfinden die Steuerung als zumindest teilweise kompliziert. Wie die Steuerelemente für die Visualisierungen verteilt sein sollen, bleibt unklar. Dies liegt teilweise an der Formulierung der Frage, deren Bedeutung oft nachgefragt wird. Konkrete Hinweise zur Aufteilung liefert die mündliche Befragung. Frage 5 klärt die spezifischen Eigenheit dieser Applikation, für jede Visualisierung ein neues Fenster zu öffnen. Dieses Konzept wird bei verschiedenen Bildbearbeitungs- und Grafikprogrammen eingesetzt, dem Autor ist aber nur ‚Gimp' (The Gimp Team 2005) bekannt, wo die einzelnen Fenster ebenfalls frei auf dem Desktop schweben und nicht in einer Arbeitsfläche liegen. Eine klare Mehrheit ist mit dieser Aufteilung einverstanden.

Dieser Evaluationsteil bietet gerade für die mündlichen Befragung viele Ansatzpunkte. Jeder Tester kennt verschiedene grafische Benutzeroberflächen und kann aus seinem eigenen Erfahrungsschatz schöpfen. Es zeigen sich teilweise grosse Unterschiede in den Meinungen, durch welche Komponenten sich das optimale Benutzerinterface auszeichnet. Es erstaunt, wie vielfältig die Ideen zur Gestaltung des Untersuchungsgegenstandes ausfallen. Andererseits gibt es Elemente, die von allen Testpersonen bemängelt werden.

Eine der Hauptkritiken betrifft die komplizierte Steuerung über das Applikationsmenu. Die Absicht ist, dadurch mehr Platz für Visualisierungsfenster auf dem Bildschirm zu schaffen. Alle Funktionen lassen sich über diese Menuleiste aufrufen und sind mit einem Tastenkürzel versehen. Sind diese Kürzel bekannt, erlauben sie eine relativ schnelle Auswahl der Abläufe. Dieses Kennenlernen ist im Umfang der Evaluation aber nicht möglich. So taucht vielfach der Wunsch auf, Schaltflächen (Buttons) direkt in das Hauptfenster zu integrieren. Mindestens die Funktionen ‚Add new frame', ‚Calculate difference frame', ‚Remove frames' und ‚Redraw frames' sollten so steuerbar sein. Eine weitere Idee ist, je nach gewünschter Funktion unterschiedliche Steuerelemente ein- und auszublenden. Dies entspricht einem Steuerungsansatz mittels Funktionsdialoge, was ein grundlegend anderer Steuerungsmechanismus als der Implementierte ist. Der Vorschlag ist sehr interessant und bietet eine grössere Flexibilität als der gewählte statische Ansatz, erfordert allerdings mehr programmiertechnische Fertigkeiten. Mehrere Gutachter äussern den Wunsch nach einer stärker dialoggesteuerten Schnittstelle. Zum Beispiel soll eine mittels Dateidialog ausgewählte NetCDF Datei direkt gezeichnet, und nicht erst in der ‚pathes'-Liste abgelegt werden. Eine weitere Vereinfachung bietet der Ansatz auch bei der Herstellung von Differenzenfenstern, denn die eingesetzte Lösung mit der doppelten Fensterliste zu deren Herstellung ist problematisch. Die Duplizierung der Liste verwirrt die meisten Testpersonen. Oft bleibt unklar, welche der Beiden für die Änderungs- und Löschfunktionen die massgebende ist, obwohl einheitlich die obere Liste die Führung übernimmt, die Untere nur zur Herstellung der Differenzenvisualisierungen gebraucht wird. Deshalb soll Letztere einen weniger dominanten Platz erhalten, z.B. als Drop-down-Liste. Die eleganteste Lösung wäre das komplette Entfernen und die Implementierung eines Differenzendialoges. Wenn die Listen beibehalten werden, ist mindestens eine Beschriftung pro Liste anzubringen.



Eine Änderung, die alle Testpersonen fordern, ist das definitive Schliessen der Visualisierung über das ‚Schliessen'-Feld in der rechten oberen Ecke des Fensters (siehe Abbildung links). Dies ist eine intuitive und gebräuchliche

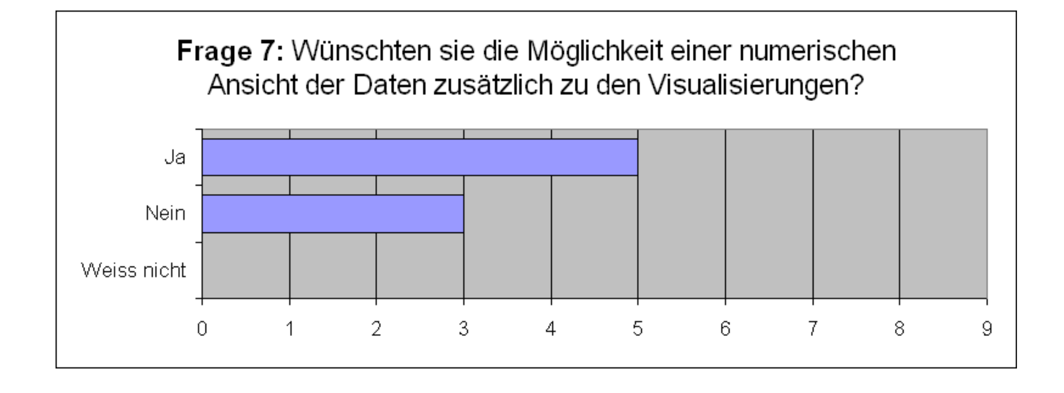
Interaktion und muss in späteren Applikationsversionen funktionieren. Nahe mit dieser Funktion verwandt ist eine Signal-Rückkoppelung vom aktiven Visualisierungsfenster zum Hauptfenster, wo der betreffende ‚Frames'-Listeneintrag markiert werden soll. Gleichzeitig sollen die aktiven Werte des Fokus-tragenden Fensters in den entsprechenden Steuerelementen erscheinen. Die Zuordnung der Visualisierungsfenster zu den korrespondieren ‚Frames'-Listeneinträgen ist generell schwierig, weil die zur Unterscheidung verwendeten Dateinamen lang sind und sich bei vielen Modellläufen je nach Konvention bei der Namensgebung stark gleich. Als Verbesserung wird eine einfache Nummerierung zur Differenzierung der Fenster vorschlagen. Ein ‚Change settings'-Dialog übernimmt sämtliche in diesem Absatz vorgestellten Funktion allerdings vollständig. Er ist eine der ersten konkreten Änderungen, die aus dieser Evaluation resultieren.

Ein Fehler beim Erneuern der Darstellungen fällt ebenfalls sehr negativ fällt. Die Darstellungen werden nicht automatisch aufgefrischt, sondern der Befehl dazu muss manuell erfolgen. Dafür fehlt ein eigener Button. Die der Funktion zugewiesene Tastenkombination ist zudem nur aktiv, wenn das Hauptfenster den Fokus trägt. Da es sich hierbei um ein Programmierproblem handelt, darf man annehmen, dass dessen Behebung in einer professionellen Anwendung keine Schwierigkeiten bereitet. Somit werden die ‚(Re)Draw'-Funktion und die damit verbundenen GUI-Objekte überflüssig.

Zwei Änderungen an den Beschriftung werden gefordert. Der ‚Zoom'-Balken zur Regelung der Darstellungsgrösse der Rasterzellen von 1x1 bis 4x4 Pixel, soll die Beschriftung ‚Sizing' erhalten. Dies mit der Begründung, dass ein echter Zoom in einem fixen Rahmen einen mehr oder weniger grossen Kartenausschnitt darstellt, während sich hier die Fenstergrössen anpassen, dabei aber immer das gesamte Raster sichtbar bleibt. Der andere Namenswechsel betrifft den ‚Color mode', der Titel für die Auswahlbox des Farbverlaufes. Der Vorschlag, ‚Color pattern' ist bereits übernommen.

Weitere kleinere Anregungen sind unter anderem: Tooltips (Erklärungsboxen) für die grafischen Elemente implementieren, die deren Funktion und die Anwendung beschreiben und so eine Soforthilfe zur Laufzeit bieten. Die Grösse der Visualisierungsfenster fixieren, da eine Veränderung daran keinen Effekt auf die Visualisierung selbst bewirkt. Der Animationsfunktion soll einen Stop-Button beigeben werden, sonst kann durch falsch gesetzte Werte ein langes Warten resultieren, weil momentan das Abspielen der Zeitschritte nicht unterbrochen werden kann. Zusätzlich zum Menu soll eine Werkzeugleiste eingeblendet werden. Der ‚Time slice'-Balken soll dorthin springen, wo er angeklickt wird.

5.2.3 Funktionalität



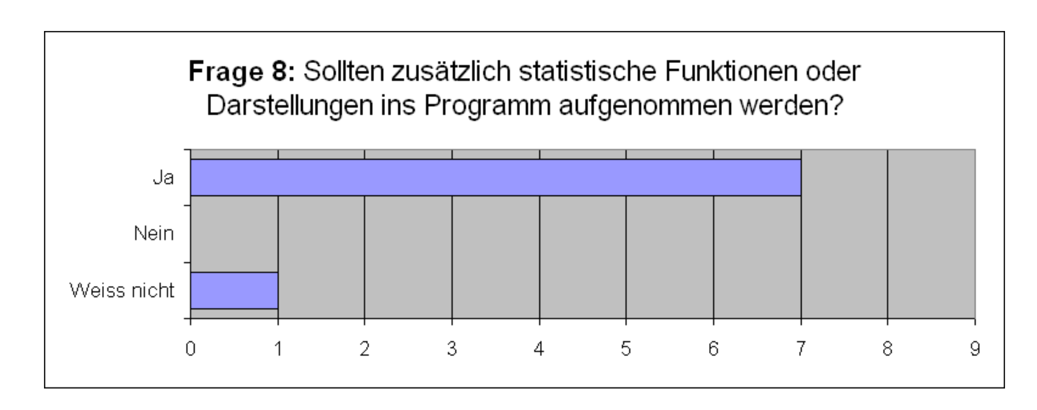


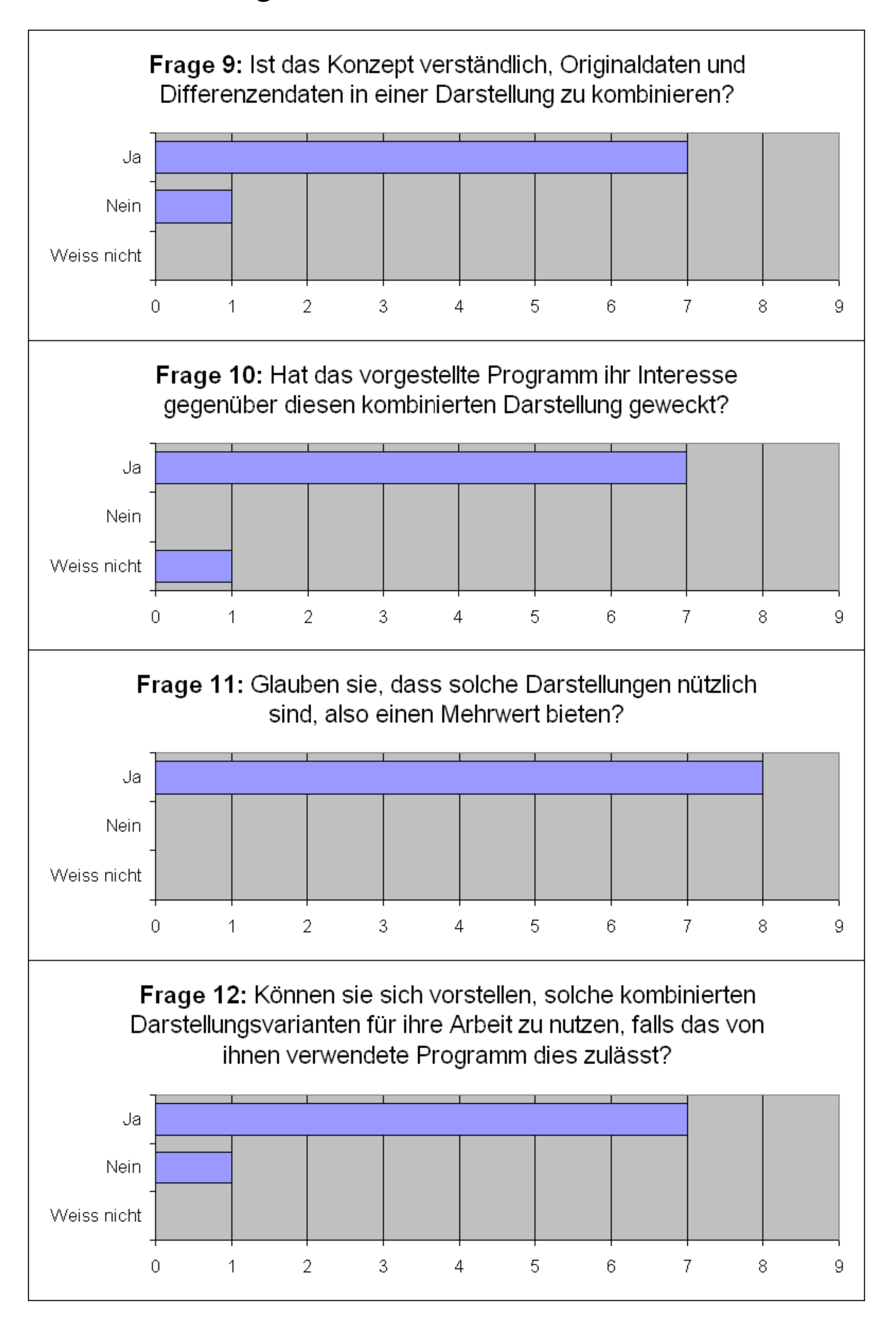
Tabelle 5-3 Auswertung der schriftlichen Fragen 7 und 8 des Fragebogens

In diesem Abschnitt schweift der Blick etwas über die Grenzen der Applikation hinaus. Zwei Themenkreise werden betrachtet, die eine Ergänzung zu den vorhandenen Funktionalitäten bieten. Frage 7 des Fragebogens spricht das Interesse gegenüber den rohen Originaldaten an (Tabelle 5-3). Es kann Situationen geben, in denen der Benutzer die exakten Werte erfahren will. Dann ist es für ihn bequemer, wenn er zu deren Ansicht mit der gleichen Software weiterarbeiten kann. Das gilt ebenso für Frage 8, die den Wunsch nach statistischen Darstellungsvariationen abklärt. Die oben abgebildete Auswertung zeigt, dass die Testpersonen grundsätzlich für mehr Funktionalitäten stimmen.

Neben den bereits in der schriftlichen Befragung erwähnten statistischen Darstellungen, nennen die Testpersonen kaum neue Funktionalitäten. Zum Einen möchten sie gerne eine numerische Angabe jener Daten, die unter dem Mauszeiger liegen, wenn dieser über einem Raster eines Visualisierungsfensters liegt. Die Anzeige soll entweder in einem eigenen Bereich ausserhalb der Zeichnungsfläche, oder direkt am Mauszeiger in einer schwebenden Box stehen. Uneinigkeit herrscht darin, ob die Datenangabe nur bei einem Mausklick auf eine Rasterzelle erscheinen soll, oder ob eine permanente Sichtbarkeit der Anzeige sinnvoller ist. Eine Kombination beider Varianten könnte beide Bedürfnisse zufrieden stellen. Bei den Differenzendarstellungen sollen zudem die Werte aller drei Fenster (Original 1, Original 2 und Differenz) angezeigt werden. Anstelle nur einer Werteangabe direkt unterhalb der aktuellen Mausposition möchten einige Befragte zusätzliche Einsicht in das nähere Datenumfeld, beispielsweise einem 5x5 Rasterzellen umfassenden Quadrat. Eine Testperson schlägt ausserdem vor, den aktuellen Wert auch mittels Pfeil auf den Wertebalken (rechts im Visualiserungsfenster) zu kennzeichnen, da die Wertzuordnung über die Farbe schwierig sei.

Ein weiterer Funktionswunsch ist das Erstellen von Transekten. Es soll die Möglichkeit geben, zwei Rasterzellen zu markieren und anschliessend einen 2-dimensionalen Querschnitt durch das Raster von Punkt eins zu Punkt zwei zu erhalten. Diese Möglichkeit bietet u.a. das Programm ncView (Pierce 2003), das ebenfalls NetCDF Dateien visualisiert. Einen Ausbau dieser Funktion ist für die Differenzendarstellung denkbar, wo der Transekt über alle 3 Datensätze berechnet und gemeinsam gezeichnet werde kann.

5.2.4 Visualisierungen



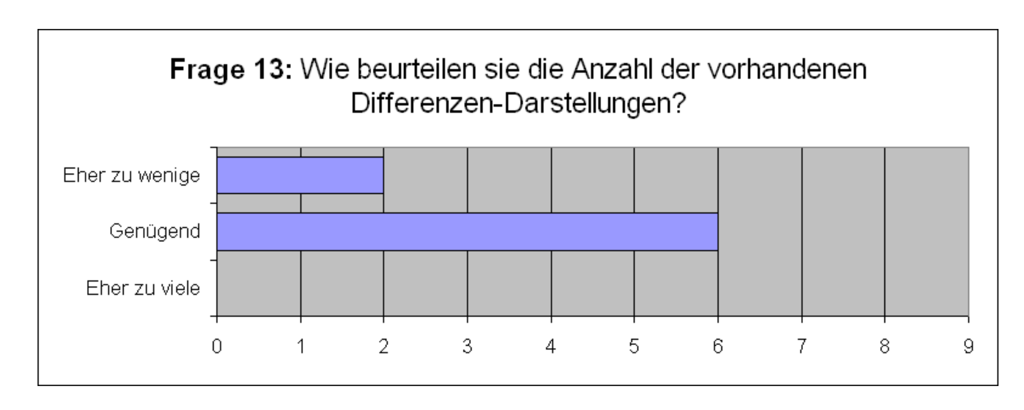


Tabelle 5-4 Auswertung der schriftlichen Fragen 9-13 des Fragebogens

Die letzten 5 schriftlichen Fragen betreffen das Kernthema dieser Diplomarbeit, die Visualisierung der Daten. Die numerische Ansicht der Resultate kann der Tabelle 5-4 oberhalb des Textes stehend entnommen werden. Während in der Befragung konkreter auf die einzelnen Umsetzungen eingegangen wird, sprechen die Fragen 9 bis 12 das Grundkonzept der Darstellungen an. In ihrer Reihenfolge bauen sie aufeinander auf, wobei Frage 9 das Verständnis des Konzeptes erkundet, Frage 10 die Neugierde der Testenden untersucht, Frage 11 den Nutzen der Darstellungen anspricht und schliesslich Frage 12 ein wenig das Marktpotiential ausloten soll. Die Freude am schönen Aussehen der Visualisierungen reicht nicht für eine gänzlich positive Bewertung. Es soll auch die Bereitschaft zum späteren Einsatz des Produktes bestehen. Die Antworten zu allen 4 Fragen sind für den Autor erfreulich. Das Konzept der Darstellungen wird verstanden und macht neugierig. Ausnahmslos alle Testpersonen gestehen den Visualisierungen einen Mehrwert zu, fast alle können sich deren Einsatz vorstellen. Frage 13 beschäftigt sich mit der Quantität der zur Auswahl stehenden Darstellungsvarianten. Eine Mehrheit findet diese genügend.

In der mündlichen Befragung ergeben sich vertiefende Einsichten betreffend den Wahrnehmungen der erstellten Visualisierungen, sowie konstruktive Vorschläge für Ergänzungen derselben oder gänzlich neuer Darstellungsarten.

Bei den ‚gewöhnlichen' Visualisierungen, die die Originaldaten unverändert darstellen, liegen die Farbton- und die Grauwerte-Farbverlaufsvarianten in der Gunst der Gutachter klar vor den beiden anderen. Sie sind deshalb beliebter, weil sie die Datenunterschiede mit ihrem breiteren Spektrum deutlicher hervorheben. Die Visualisierungen, die mit den Farbvariablen Farbsättigung und Farbwert arbeiten, zeigen dagegen sehr schwache Farbübergänge. Vor allem Letztgenannte ist sehr dunkel und erschwert die leichte Wertedifferenzierung. Allerdings empfindet eine Testperson sie als ästhetisch ansprechend, „wie ein Bild, dass man auch an die Wand hängen kann“. Es wird der Vorschlag geäussert, das Graustufenschema zu invertieren, damit das kartografische Gebot ‚grössere Werte, dunklere Farben' erfüllt ist. Als störend empfinden einige, das in der Farbton-Darstellung der 0-Wert rot ist, lieber wäre ihnen weiss, oder gar keine Farbe, also grau wie der Hintergrund. Eine andere Begrenzung der Farben (z.B. von Gelb bis Blau) ist ein weiterer Vorschlag. Die Rot-Blau-Visualisierung, die zwei Originaldatensets miteinander vermischt, gefällt manchen recht gut. Vor allem die verschiedenen Umrisse der abgedeckten Datenbereiche seien deutlich sichtbar. Vereinzelt fällt die Frage, weshalb diese Visualisierung bei den Differenzendarstellungen eingeordnet ist, obwohl die Differenz nicht direkt dargestellt wird. Der Umstand lässt sich damit begründen, dass einerseits der Unterschied so ersichtlich wird, andererseits diese Einordnung durch die programmtechnischen Datenbezüge einfacher ist.

Als nächstes folgen die Aussagen über die Differenzendarstellungen. Die einfacheren Dithering- bzw. Schattierungsdarstellungen erhalten gemeinhin eine gute Bewertung. Die gebräuchlichsten Adjektive zu ihrer Beschreibung sind klar, einfach und intuitiv. Je nach Testperson ist mal die Dithering-, mal die Schattierungsvisualisierung beliebter, mit einer knappen Mehrheit für die Dithering-Variante. Eine Testperson findet die Legende zu dieser Visualisierung unverständlich, weil sie die Differenz nicht auch als Punktraster zeichnet, sondern als sich verbreiternder schwarzer Balken. Dies soll die stärkere Dunkelfärbung der Rasterzellen symbolisieren, wird aber als zu abstrakt gesehen.

Die experimentellen Block- bzw. Gitterdarstellungen, deren Berechnung mittels Quadtree-Algorithmus geschieht, finden bei den Testern schwachen Anklang. Sie sind schwer nachvollziehbar, es bleibt lange unklar, was die Abbildungen bedeuten. Der Gitter-Ansatz findet eher Befürworter, erhält aber Kritik durch die verwendete Maschenweite. Für viele intuitiver (und kartografisch korrekter) wäre eine umgekehrte Anwendung der Maschen, so dass bei kleinen Differenzen grosse Gitterzellen und bei grossen Differenzen kleine Gitterzellen gezeichnet würden. Damit entspräche die Symbolik der Differenz den Dithering und Schattierungsvisualisierungen. Eine einfache Umsetzungsmöglichkeit ist, die berechnete Differenz von der maximalen Differenz zu subtrahieren, womit eine Art Kehrwert kalkuliert wird, mit dem der gleiche Algorithmus wie jetzt verwendet werden kann. Eine Testperson äussert die Idee, nicht nur die Maschenweite des Gitters anzupassen, sondern zusätzlich mit der Helligkeit der Maschen zu arbeiten, wobei kleinere Differenzen wiederum heller gezeichnet werden. Ist der Unterschied 0, die gegenübergestellten Modelle also deckungsgleich, soll ganz auf das Zeichnen das Gitters verzichtet werden. Die Blockdarstellung gilt als die am schwierigsten zu lesende Visualisierung. Die verfolgte Intention, unsichere Daten als tiefer aufgelöste Einheiten zu präsentieren, wird für gut befunden. Allerdings entsteht dadurch der Eindruck einer Klassierung. Dieser Eindruck ist teilweise richtig, den die Differenz wird tatsächlich in Klassen unterschiedlicher Blockgrössen eingeteilt. Eine stetige Änderung ist mit diesem Algorithmus nicht erzielbar.

Bei allen Differenzendarstellungen, deren Basis durch ein Originalraster gebildet wird, soll die Auswahl des Originalfensters zur Verfügung stehen. In der jetzigen Umsetzung verwendet die Applikation automatisch Frame 1. Das ist der Eintrag, der bei der Erzeugung des Differenzenfensters in der oberen ‚Frames'-Liste ausgewählt ist.

Das Konzept der wählbaren Farbklassierungen soll erweitert und flexibilisiert werden. Ein vielfach geäusserter Wunsch ist die Möglichkeit der Definition eigener Farbschemen. Dies ist ein Punkt mit viel Ausbaupotential. Offen ist die Frage, wie das dazugehörende Werkzeug gestaltet sein soll, ob als grafischer Dialog, oder als Textfile, das beim Programmstart geladen wird. Ersteres ist intuitiver und einfacher für den Nutzer, da er auf entsprechende Erfahrung aus anderen Programmen zurückgreifen kann, aber mit mehr Kodieraufwand auf der Entwicklungsseite verbunden. Letzteres böte die Möglichkeit einer permanenten Sicherung der erstellten Farbschemen sowie deren einfachen Erweiterbarkeit. Eine einfache Deklaration nicht-linearer Farbgebungen ist so ebenfalls realisierbar, indem nur die Eckdaten der Farbzwischenstufen beschrieben werden, während die Applikation die feinen Zwischenstufen automatisch interpoliert. Ein gravierendes Problem beinhalten solche Textdateien allerdings: Sie sind wenig intuitiv; das Editieren-Müssen einer programmexternen Datei könnte unerfahrenere Benutzer vom Gebrauch der Funktionalität abschrecken.

Den Testenden wird kurz das Setzen von Minimum und Maximum des Farbverlaufes präsentiert, um ihnen einen Eindruck davon zu geben, wie sich dies auf die Visualisierungen auswirkt. Ein Test dieser Funktionen findet dagegen nicht statt, da sie sich in einem experimentellen Entwicklungsstadium

befindet. Der Mechanismus gefällt jenen Personen, die etwas mehr Zeit haben, sich damit auseinander zusetzen. Sie argumentieren, dass sich damit die Darstellungen fein modulieren lässt.

Die Legende neben der Rasterdarstellung zeigt das gewählte Farbmuster/Differenzenmuster (breiter Balken) und den Wertebereich, den die Daten einnehmen (schmaler schwarzer Balken daneben). Wenn keine eigenen Limiten zur Verlaufsdarstellung eingestellt werden, sind die beide jeweils gleich lang. Da in den kombinierten Differenzendarstellungen zwei Skalenpaare nötig sind, ergibt dies maximal 4 Balken. Einige Tester verstehen diese Legende nicht, da zu viele Balken angezeigt werden. Die Beziehung der Balken eines Paares untereinander ist verworren. Zudem ist schlecht, dass der Balken für die Anzeige der Spannweite der echten Werte über das Fenster hinausgezeichnet wird, weil der Balken für die Visualisiertenwerte eine konstante Grösse behält. Besser ist, Letzteren je nach Einstellungen auch anzupassen. Dies zeigt, wie eng verwoben die Längen der Balken mit dem direkt oberhalb angesprochenen Thema ‚Grenzwerte für einem Farbverlauf setzen' sind.

Grundsätzlich soll an allen Visualisierungen festgehalten werden, da die Auswahl an Darstellungsvarianten mehr Betrachtungsmöglichkeiten offen lassen. Damit lässt sich mit den Daten spielen, der Betrachter erhält vielschichtigere Einblicke und so vielleicht vermehrte Denkanstösse bei deren weiterer Nutzung. Je nach Datenverteilung kann der Nutzen der verschiedenen Visualisierungen schwanken. Um dies ausführlich zu testen fehlen aber genügend unterschiedliche Daten und die Zeit dazu.

6 Diskussion

In diesem Kapitel werden die verschiedenen Teile der Diplomarbeit diskutiert. Dabei werden die in der Einleitung formulierten Fragen als Gerüst für die Unterkapitel verwendet und die unterschiedlichen Gesichtspunkte der Arbeit danach untersucht. Als Abschluss wird die verwendete Hypothese verifiziert.

6.1 Vermittlung von Unsicherheiten

Frage Nummer I ist auf die grundsätzlichen Repräsentierungsmöglichkeiten von Rasterdaten ausgelegt. Die Einschränkung auf Rasterdaten erfolgt mit Blick auf die Testdatenbasis. Die Frage lautet:

I) „Wie kann man Unsicherheiten in Rasterdaten darstellen?“

Diese Frage wird in Kapitel 2 ‚Grundlagen’ untersucht. Dabei wird zu Beginn der Betrachtungen der Begriff ‚darstellen’ nicht auf die visuelle Komponente beschränkt, sondern es werden alle denkbaren Unsicherheitsrepräsentationen betrachtet. Grundsätzlich darf keiner der menschlichen Sinne von vornherein ausgeschlossen werden. Bei der gewählten Versuchsanordung können jedoch der Geschmacksinn, der Geruchssin und der Berührungssinn nicht angesprochen werden. Die Technik zur Erzeugung künstlicher Gerüche ist noch zu wenig weit fortgeschritten, um genutzt zu werden.

Der Einsatz des Gehörsinns ist mit den heutzutage verwendeten Desktop-Computern und ihren technischen Mitteln möglich. Dabei wird die Datenqualität unterschiedlichen Toneffekten zugeordnet. Anstelle der Modulation der Töne kann auch die Lautstärke variiert werden. Die akustische Vermittlung der Datenqualität wird aus folgenden Gründen weggelassen. Erstens ist dies ein sehr eingeschränkter Untersuchungsgegenstand, der weniger Variationsmöglichkeiten bietet als Visualisierungen. Zweitens kann nur jeweils ein Unsicherheitswert gleichzeitig akustisch symbolisiert werden. Entweder ist dies der Durchschnitt aller Datenwerte, oder jener für eine Zelle. Das Ziel der Arbeit ist aber eine schnelle Übersicht über die differierenden Unsicherheiten eines ganzen Datensatzes zu bieten.

Somit bleiben die visuellen Darstellungsmöglichkeiten übrig und bilden den Fokus der Arbeit. Visualisierung spricht die sichtbare Informationsweitergabe an.

Dafür sind mehrere Vorgehensweisen möglich. Eine zu bestimmende Grundsatzentscheidung ist, ob die Daten und die Qualitätsdaten getrennt in zwei separierten Visualisierungen oder in einer vereint dargestellt werden. Der Autor entschied sich für die zweite Möglichkeit. Ihr Konzept ist neuer und sie kann mehr Informationen auf kleinerem Raum anzeigen. Ausserdem wirkt es realitätsnaher, beide Daten miteinander zu verbinden und gleichzeitig darzustellen, so dass die korrespondierenden Teile beieinander stehen. Bei getrennten Darstellungen ist der Benutzer gezwungen, die zusammengehörenden Daten selbst zu finden. Dies kann zu Interpretationsfehlern führen.

Es wird eine grafische Sprache gewählt, mit der Bildinhalte analysiert und in Einzelteile zerlegt werden können. Ein sehr flexibles Beschreibungsgerüst bietet das Prinzip der Grafikvariablen, das erstmals durch Bertin 1967 (in Blok 2000) beschrieben wurde. Es unterteilt die Visualisierungen in

klare Komponenten. Zudem kann diese Beschreibung auf unterschiedliche Weise erweitert werden. Dies kann ein Perspektivenwechsel in der Betrachtung und Analyse der Grafiken sein, oder eine Zerlegung der Darstellungen in höhere Abstraktionsgrade. Diese Argumente überzeugen den Autor, mit dem Konzept der Grafikvariablen zu arbeiten.

Durch den angesprochenen Perspektivenwechsel werden Bilder nicht mehr nur nach ihren Grundelementen aufgeteilt. Andere Kriterien nehmen eine objekt- oder aktionsbezogene Einteilung der Bildelemente vor. Zudem erweitert der Einbezug der zeitlichen Dimension das Darstellungs-spektrum. Damit können die Vorteile des Computers gegenüber statischen, gedruckten Kartenerzeugnissen eingesetzt werden. Die Metapher der Grafikvariable passt dann nicht mehr und es wird von Visualisierungstechnik gesprochen. Sinnvollerweise wird in dieser Arbeit keine feste Grenze zwischen diesen Bereichen gezogen, da ihr Übergang fliessend ist. Die erfolgte Einteilung nach Bertinschen Grafikvariablen, anderen Grafikvariablen und Visualisierungstechniken ist ein strukturierendes Element und hat für die praktische Umsetzungen wenig Relevanz.

6.2 Grafikvariablen und Visualisierungstechniken

Frage Nummer II betrifft die verwendbaren grafischen Elemente und ist folgendermassen formuliert:

II) „Welche grafischen Variablen – beziehungsweise Visualisierungstechniken – sind für diese Darstellungen besonders geeignet?“

Ein grosser Teil des Grundlagenkapitels beschreibt die unterschiedlichen Grafikvariablen und deren Verwendung zur Unsicherheitsvisualisierung. Dabei werden diese Grafikvariablen nach ihrer Herkunft unterteilt. Bei der Betrachtung der Visualisierungseignung gibt es keinen Unterschied, von wem oder wann sie zum ersten Mal beschrieben wurden. Trotzdem ist eine leichte Tendenz festzustellen, dass die am längsten bekannten Grafikvariablen gleichzeitig am meisten Visualisierungspotenzial mitbringen. Dabei sei auf den Farbton, den Farbwert, die Textur und die Farbsättigung hingewiesen, die ausgezeichnete Möglichkeiten hierfür bieten. Die Farbsättigung wurde bereits von Bertin erwähnt, obwohl er sie nicht als eigenständige Varible anerkannte. Auch der Einsatz einiger neuerer Grafikvariablen empfiehlt sich, insbesondere der Transparenz und der (Kanten)Schärfe. Allerdings kann bei deren Darstellung und Beschreibung auf die älteren Grafikvariablen zurückgegriffen werden. Sie lassen sich als zusammengesetzt Grafikvariablen betrachten. Es erstaunt nicht, dass die länger bekannten, weniger komplexen und abstrakten Variablen mehr und flexiblere Einsatzmöglichkeiten aufzeigen. Sie wurden nicht zuletzt wegen ihrer Einfachheit als erste entdeckt.

Die den einzelnen Grafikvariablen angefügten Kombinationstafeln sind so ausführlich und vollständig wie möglich gezeichnet. Trotzdem sind einige Weglassungen nicht vermeidbar. Dies bedeutet nicht, dass diese nicht funktionierten. Vielmehr passen die nötigen Farbtafeln nicht in das verfolgte Darstellungskonzept, oder waren in der verwendeten Schematik schwer realisierbar. Teilweise gleichen sich die erzeugten Farbtafeln so sehr, dass man sie für eine doppelte Wiedergabe der selben Farbtafel halten könnte. Als Beispiel lassen sich die Graustufendarstellung mit der Farbwert und Farbsättigung vergleichen: Alle drei sind Hell-Dunkel Übergänge, die beiden letzteren in einem frei wählbaren Farbton.

Es sollte nicht vergessen werden, dass die Auswahl der Grafikvariablen, insbesondere der jüngeren, einem subjektiven Einfluss unterworfen ist. In gewissen Fällen ist es schwierig zu entscheiden, ob es sich bei einer Bildkomponente um eine Grafikvariable im engsten Sinne des Wortes handelt, oder eher

um eine zusammengesetzte ‚Grafikvariable' höherer Ordnung. Von diesen komplexeren Grafikvariablen hin zu den verschiedenen Visualisierungstechniken, die wiederum sehr unterschiedliche Komplexitäten und Anforderungen aufweisen, ist es ein kleiner Schritt. Die vorgenommene Einteilung in die drei grossen Gruppen ‚klassische Grafikvariablen', ‚neuere Grafikvariablen' und ‚Visualisierungstechniken' kann etwas zu statisch erscheinen. Realistischer ist wohl die Betrachtung als Kontinuum, in der die genaue Einteilung zugunsten der Einsatzmöglichkeiten zurücktritt. Andererseits ist die Einteilung im Hinblick auf die dadurch gewonnenen grössere Übersichtlichkeit sicher vertretbar.

Wie bereits erwähnt, werden einige Visualisierungstechniken aus verschiedenen Gründen weggelassen. Darunter fallen schwer verständliche Konzepte, Wiederholungen bereits bekannter Techniken unter anderem Namen (teilweise leicht modifiziert) und nicht einsetzbare Konzepte. Es lässt sich nicht ausschliessen, dass manche davon mit einem anderen Ansatz als in dieser Diplomarbeit verwendet, sinnvoll eingesetzt werden können. Somit bietet sich ein interessantes Feld für weitere Forschungsarbeiten an.

6.3 Anforderungen an die Applikation

Frage Nummer III ermittelt eine geeignete praktische Umsetzung der Visualisierungen.

III) "Was sind die Anforderungen an eine Applikation diesen Typs?"

Da die verwendeten Betriebssysteme und Sprachen die Art der Implementierung einer Applikation beeinflussen, werden zunächst diese Komponenten diskutiert. Im Nachhinein betrachtet erfolgte die Vorbereitung der Applikationsimplementierung zu wenig systematisch. Die Wahl von Linux als zugrunde liegendes Betriebssystem sowie der Programmiersprache C++ mit dem qT Framework verzögerten die Umsetzung einige Zeit. Beide Umgebungen waren dem Autor noch wenig bekannt. Gerade die Verwendung von Linux verlangt einem Windows gewohnten User einiges an Umstellung ab. Persönlich wird die Erweiterung der Programmierfertigkeiten auf eine neue Sprache als sehr positiv gewertet, trotz des Mehraufwands gegenüber einer Implementierung in einer bereits bekannten Programmiersprache wie Java. Java bietet ausserdem den Vorteil einer besseren Portierbarkeit auf andere Betriebssysteme. Dies ist mit dem Einsatz das qT Frameworks nicht ohne Kosten möglich, da nur die Linux Umsetzung davon gratis zugänglich ist. Zudem scheint Java mit fortschreitendem Reifegrad C++ als die beliebteste objektorientierte Programmiersprache zu verdrängen. Ein Grund dafür mag die mittlerweile ebenso hohe Laufzeitgeschwindigkeit sein, ein anderer die Stabilität. Zudem verfügt Java mit seiner grossen Sammlung an vorgefertigten Klassen über einen starken Vorteil gegenüber den sehr schlank gehaltenen C++ Kernklassen. Es soll darauf hingewiesen werden, dass eine genaue Evaluation der zu verwendenden Softwaremittel bereits vor dem Schreiben der ersten Zeile Programmcode einen nicht zu unterschätzenden späteren Zeitgewinn einbringen kann. Der entwickelte Prototyp wird allerdings nicht grundsätzlich in Frage gestellt. Die verwendete technische Umgebung erfüllt ihren Zweck sehr gut. Alle angedachten Funktionen lassen sich mit sinnvollem Aufwand umsetzen. Die Applikation läuft stabil, einige Funktionen sogar bedeutend schneller als erwartet. Dies ist besonders bei der Animationsfunktion zu beobachten: Bei mehreren geöffnet Originaldaten- und Differenzendaten-fenstern läuft diese sehr flüssig. Dabei kann die rechenintensive Quadtree-Differenzendarstellung ohne spürbaren Geschwindigkeitsverlust eingesetzt werden.

Da kaum auf Erfahrungen betreffend einer Applikation diesen Typs zurückgegriffen werden kann, erfolgte der Entwicklungsvorgang sehr experimentell. Die Applikation wurde mehrmals umgestellt. Einerseits änderten sich die verwendeten Daten im Laufe der Implementierung, andererseits wurden bestehende Klassen in mehrere aufgeteilt. Die eingesetzte Klassenstruktur erweist sich in ihrer jetzigen Form als ausreichend flexibel. Trotzdem existieren noch einige Verbesserungsideen, die einen klareren und flexibleren Programmaufbau bewirken würden.

Die Ableitung der verschiedenen Visualisierungsklassen von einer zentralen virtuellen Hauptklasse, die sämtliche Zugriffsmethoden bereits deklariert, ermöglicht das kompakte Speichern aller Visualisierungsinstanzen im gleichen Klassencontainer. Eine Erweiterung mit anderen Visualisierungsklassen ist so leicht programmierbar. Ein identisches Vorgehen bietet auch für die Zeichnungsklasse (Painterklasse) Vorteile. Zurzeit sind alle Zeichnungsmethoden in einer einzigen Klasse vereint, wodurch diese ziemlich gross und unübersichtlich ist. Hier wäre ein modularer Aufbau dringend ratsam. Vorstellbar wäre eine Zeichnungsklasse für jede einzelne Visualisierung, die ebenfalls von einer virtuellen Hauptklasse die Methodenschnittstellen erbt. Der Quellcode jeder Zeichnungsklasse würde dadurch sehr viel kürzer und einfacher wartbar. Die jetzige Lösung ist mit dem dynamischen Wachsen der Applikation zu erklären. Ursprünglich bestand die Idee, alle Visualisierungsvarianten mittels einer oder zwei Methoden zu realisieren. In diesen hätte dann Schlaufenbedingungen die Auswahl der zu verwendenden Zeichnungsparameter geregelt. Schon bei wenigen Visualisierungsoptionen zeigte sich allerdings, dass sich die Methode zu stark verschachtelt. Sie war für Aussenstehende kaum noch nachvollziehbar, weshalb auf die Variante mit vielen Methoden gewechselt wurde. Dieser Punkt unterliegt einer Designschwäche, die in einer späteren Programmversion behoben werden muss. Die GUI-Klassen sind für den Prototyp ausreichend. Sämtliche implementierten Funktionen sind mit vertretbarem Aufwand einsetzbar. Die Evaluation hat gezeigt, dass weitere Funktionen gewünscht werden, die im Evaluationsteil dieses Kapitels besprochen werden.

Die Implementierung flexibler Rastergrössen hat sich als schwierig erwiesen. Der entwickelte Prototyp vermag im derzeitigen Stadium nur eine fixe Rastergrösse darzustellen. Jede Änderung an den Ausmassen des Raster bedingt eine Recompilierung der Applikation. Dies kann mit wenig Aufwand durchgeführt werden, indem im Programmcode die verwendeten statischen Variablen mit den korrekten x- und y-Koordinaten überschrieben werden. Allerdings bedingt dies, dass der Benutzer der Applikation diesen Vorgang versteht und bereit ist, diese Manipulation am Quellcode vorzunehmen. Auf einige potentielle Benutzer mag dies abschreckend wirken. Zudem sind bei der Verwendung vieler unterschiedlicher Rastergrössen zu viele Applikationsversionen zu benutzen. Bei der gleichzeitigen Darstellung dieser unterschiedliche Rastergrössen wird jeweils eine eigene GUI gestartet. Dadurch wird viel Bildschirmfläche beansprucht und die Programmsteuerung schnell unübersichtlich. Ausserdem wird so der Speicher- und Rechenbedarf unnötig erhöht.

Die Installation der Applikation ist mit grossem Aufwand verbunden. Neben der sehr grossen qT-Klassenbibliothek, die nicht standardmässig auf allen Linuxsystemen vorhanden ist, muss auch das NetCDF-Packet heruntergeladen und installiert werden. Anschliessend erfolgt die manuelle Compilierung der entwickelten Applikation. Ein Benutzer, der dieses Prozedere vollzieht, dürfte demnach mit dem Anpassen des Quellcodes für die eigenen Raster kaum Mühe haben.

6.4 Bewertung der Umsetzung

Frage Nummer IV befasst sich mit der Evaluation der Applikation.

IV) „Wie urteilen Testpersonen über das Programm und die Darstellungen?“

Da viele der gewonnen Erkenntnisse die Evaluation selbst betreffen, werden diese Resultate als erstes diskutiert.

Eine wichtige Erkenntnis, die unmittelbar den gesamten Designprozess sowie die Implementierung betrifft, ist die Notwendigkeit einer vorgängigen Informationsbeschaffung zu den von den Benutzern gewünschten Funktionen, Visualisierungsoptionen und grafischen Interaktionsmöglichkeiten. Unter letztem Punkt sind die Gestaltung der Oberflächenelemente sowie die Anordnung und Abfolge dieser Elemente zu verstehen. Die nachträglich durchgeführte Evaluation ergab sehr viele gute Ideen und Anregungen zu allen aufgeführten Aspekten. Viele dieser Wünsche sollen in zukünftigen Versionen dieser Applikationen integriert werden. Manche vom Entwickler gefällte Entscheidung muss direkt auf die Bedürfnisse der Endnutzer zugeschnitten sein. Mittels einer Vorstudie vor Beginn der Implementierung hätten bereits einige dieser Anregungen erkannt und integriert werden. Eine solche Vorstudie wurde jedoch nicht durchgeführt. Die Wahl einer nachträglichen Befragung anhand des entstandenen Prototyps geschah in der Annahme, dass erst mittels konkreter Umsetzung eine zielgerichtete Befragung möglich ist, was sich wiederum als teilweise richtig erwiesen hat. Gewisse Probleme und Entscheidungsfragen traten erst während der Implementierung auf und konnten nicht von Anfang an antizipiert werden. Mit der Umsetzung und der Befragung wurde so viel Wissen gesammelt, dass ein komplett neues Design der Applikation in Betracht gezogen wird.

Das Ziel der Befragung war ein möglichst grosses Spektrum an Beobachtungen und spontanen Äusserungen zu erhalten, um daraus Ideen für die Weiterentwicklung der Software zu bekommen. Weiter sollten mögliche Designfehler der GUI erkannt werden, um diese später zu korrigieren. Beide Ziele wurden erreicht. In den Gesprächen wurde eine Vielzahl von Anregungen gesammelt und soll in spätere Applikationsversionen einfliessen. Die Anzahl der schriftlichen Fragen wurde möglichst knapp gehalten, da die Evaluation auf eine kleine Testgruppe beschränkt wurde. Bei wenigen Testpersonen vermittelt ein qualitatives Gespräch mehr Informationen als ein standardisierter Fragebogen. Dieser gibt bei einer kleinen Stichprobe nur einen ungefähren Eindruck über die Meinungsrichtung, da für die statistische Auswertung die Mindestgrösse einer Stichprobe nicht erreicht wird. Deshalb besteht der Fragebogen aus allgemeinen Fragen, die den Gesamteindruck der Applikation erfassen sollen.

Die Antworten im Fragebogen sind sehr zufriedenstellend ausgefallen. Die Testenden verstehen das Konzept der Visualisierungen. Dies wird durch die mündliche Befragung bestätigt. Sie attestieren den Darstellungen einen Zusatznutzen. Der Autor erhielt viele positive Äusserungen zu den erzeugbaren Darstellungen. Die Testpersonen können sich vorstellen, die Visualisierungen für ihre Arbeit selber anzuwenden. Diese Frage ist für einige sehr hypothetisch, da sie im Alltag sehr selten oder nie Datenvisualisierungswerkzeuge einsetzen. Trotzdem zeigen die Antworten, dass das Interesse für die Darstellungsart vorhanden ist. Dies wird durch die direkte Frage nach dem Interesse bestätigt. Es zeigt, dass die vorgestellten Darstellungen die Neugierde der Testpersonen wecken. Dies ist für weitere Forschungsarbeiten an der Unsicherheitsvisualisierung vielversprechend und zeigt, dass nicht nur unter Spezialisten ein Bedürfnis nach stärkerer Beachtung der Datenqualität besteht.

Es ist angebracht, mit einer verbesserten Applikation eine gründlichere Studie durchzuführen. Dazu soll der Fragebogen mit zusätzlichen Fragen ausgebaut und verfeinert werden. Um die Evaluationsergebnisse statistisch fundieren zu können, muss die Testgruppe vergrössert und breitere Auswahl an Testenden mit unterschiedlichem Wissens- und Interessenstand gewählt werden.

In der jetzigen Form der Evaluation ist eine klare positive Bewertungstendenz festzustellen. Allerdings soll die Aussagekraft nicht überbewertet werden. Beim Aufbau des Fragebogens wurde nicht darauf geachtet, ob sich die Fragen gegenseitig beeinflussen. Suggestivfragen wurden so weit als möglich vermieden, können trotzdem im Zusammenhang mit der mündlichen Befragung nicht ausgeschlossen werden.

6.5 Fazit

Zum Schluss des Kapitels soll die Hypothese verifiziert werden. Diese wurde in Kapitel 1 ‚Einleitung' folgendermassen formuliert:

„Die optische Sichtung von Resultatdaten unterschiedlicher Modellrechnungen ermöglicht einen raschen Überblick über deren Unterschieden respektive Unsicherheiten. Bei der Visualisierung der Daten sollen die Stärke und die Verteilung der Unsicherheiten deutlich hervortreten und intuitiv verstanden werden."

Die Hypothese hat sich in dieser Diplomarbeit als zutreffend erwiesen. Die Evaluation zeigt, dass die Testpersonen die Unterschiede in den verschiedenen Modellen durch die verwendeten Visualisierungen tatsächlich wahrnehmen. Je nach gewählter Darstellungsart werden die Unsicherheiten unterschiedlich schnell erkannt. Mit den besten Visualisierungsverfahren sind die Differenzen in sekundenschnelle erfasst und können interpretiert werden. Die beiden beliebtesten Visualisierungen sind die Ditheringdarstellung und die Schattierungsdarstellung, die von den Testpersonen eine sehr gute Bewertung erhalten. In diesen Darstellungen werden die Gebiete mit grösseren Differenzen leicht von jenen mit kleineren Abweichungen unterschieden. Die Gebiete mit den hauptsächlichen Differenzen sind klar abgehoben, wodurch die Akzentuierung von ‚Problemzonen' gelingt. Die quantitative Unterscheidung der Unsicherheit ist schwieriger. Die Applikation bedarf in diesem Punkt weiterer Verbesserungen. Das intuitive Verständnis ist bei den beiden erwähnten Visualisierungen sehr hoch. Alle Testpersonen verstehen schnell, was dargestellt wird. Voraussetzung dafür ist eine kurze grundlegende Einführung in die Thematik der Darstellungen und des Inhalts der visualisierten Daten.

7 Schlussfolgerungen und Ausblick

Dieses Kapitel fasst die angewandte Vorgehensweise zusammen und stellt die gewonnenen Erkenntnisse vor. Als Abschluss der Diplomarbeit werden einige Anregungen für weitere Forschungsarbeiten präsentiert.

7.1 Überblick

Diese Diplomarbeit untersucht die Unsicherheitsvisualisierung in zwei Hauptbereichen. Zum einen liefert sie einen Beitrag zur Analyse des Einsatzes von grafischen Grundbausteinen, den Grafikvariablen. Zum anderen werden einige theoretisch erläuterte Grafikvariablen-Kombinationen in einer eigenen Applikation umgesetzt und anschliessend mittels qualitativer Evaluation getestet.

In Kapitel 1 ‚Einleitung' wird die Motivation und der Hintergrund dieser Diplomarbeit umrissen. Der praktische Bezug zu wirtschaftlichen und politischen Entscheidungsfindungen wird dargelegt. Weiterhin existieren viele offene Fragen in Bezug auf Unsicherheitsvisualisierungen. Die als Forschungsgrundlage aufgestellte Hypothese und die daraus abgeleiteten Fragen werden ebenfalls in diesem Kapitel erläutert.

In Kapitel 2 ‚Grundlagen' werden wichtige Begriffe wie Geovisualisierung und Unsicherheit erläutert. Bezüglich der Unsicherheit findet sich ein weites Spektrum an Interpretationen und Definitionen. Es wird aufgezeigt, in welchen Stadien des Datenverarbeitungsprozesses Unsicherheiten auftreten können. Grundsätzlich ist kein Verarbeitungsschritt unproblematisch. Als Beispiel für die Modellierung von Unsicherheitsdaten wird die Monte Carlo Simulation beigezogen und vorgestellt. Im Anschluss daran werden die Grundbausteine grafischer Darstellungen präsentiert, die sogenannten Grafikvariablen. Diese werden in die klassischen sieben Bertinschen Grafikvariablen und die später hinzugefügten aufgeteilt. Das Potential jeder einzelnen Grafikvariable zur Unsicherheitsvisualisierung wird untersucht. Der Schwerpunkt liegt dabei bei Rasterdaten. Schliesslich werden Visualisierungstechniken vorgestellt und ebenfalls nach Einsatzmöglichkeiten in der Unsicherheitsvisualisierung analysiert. Der Abschluss des Kapitels bildet eine tabellarische Übersicht über die behandelten Grafikvariablen und Visualisierungstechniken. Darin wird die Bewertung zum Einsatz in der Unsicherheitsvisualisierung kompakt präsentiert und die in der Applikation verwendeten Komponenten gekennzeichnet.

Kapitel 3 ‚Umsetzung' beschreibt die Umsetzung einiger der theoretisch erarbeiteten Visualisierungskonzepte in einer eigenen Applikation. Zuerst wird der technische Hintergrund der Anwendung erläutert. Dabei wird das verwendete Betriebssystem Linux, die eingesetzte Programmiersprache C++ mit dem qT framework und deren Eigenheiten vorgestellt. Nach der Systemumgebung wird die implementierte Applikation erläutert. Die Klassenhierarchie als XML-Diagramm wird gezeigt und in Worten beschrieben, so dass die Struktur der Applikation aufgeschlüsselt wird. Anschliessend erfolgt die Vorstellung der Grafischen Benutzeroberfläche (GUI) und den Funktionalitäten der Applikation. Diese leiten über zum Hauptteil des Kapitels, den implementierten Visualisierungsoptionen, die in die verschiedenen Visualisierungstechniken unterteilt sind. Begonnen wird mit der gleichzeitigen Darstellung mehrerer statischer Visualisierungsfenster, die den direkten Vergleich der unterschiedlichen Modelle und der Unterschiedsraster ermöglichen. Im

Anschluss daran werden die Darstellungsoptionen für die Ansichten von Einzeldaten aufgelistet. Schliesslich erfolgt die Beschreibung der Unsicherheitsvisualisierungen. Die Erläuterungen zu den Darstellungen beinhalten jeweils einen Beschrieb ihres Aufbaus. Der technische Hintergrund wird erklärt und die dazu verwendeten Grafikvariablen benannt. Die Applikation ist so konzipiert, dass weitere Visualisierungsvarianten mit wenig Programmieraufwand hinzugefügt werden können.

Kapitel 4 ‚Evaluation' ist dem Applikationstest gewidmet. Bei der gewählten Testanordnung, ein ‚Structured Walkthrough' ohne Vorbereitung der Testpersonen, handelt es sich um eine offene Interviewform. Alle acht Testpersonen stammen aus dem geografischen Institut Zürich. Sie bilden die Testgruppe, mit der in Einzelsitzungen die Evaluation durchgeführt wurde. Die Evaluation selbst ist in einen vierstufigen Prozess unterteilt. Die Testpersonen erhalten als Erstes das Manual zu lesen. Dieses ist als Hypertext-Dokument verfasst und zum Lesen in einem Internet-Browser konzipiert. Als zweiter Schritt werden die Testpersonen gebeten, einige vordefinierte Aufgaben in der Applikation nachzuvollziehen. Dabei entsteht gleichzeitig eine Diskussion zwischen dem Testleiter (hier der Autor) und den Testpersonen. Schritt drei besteht aus dem Ausfüllen eines standardisierten Fragebogens, bevor als letzter Schritt eine mündliche Befragung stattfindet. Die Antworten des Fragebogens werden in einer tabellarischen Form in themenspezifisch geordneten Kapiteln präsentiert, in denen direkt anschliessend die Rückmeldungen aus der mündlichen Befragung zusammengetragen werden.

In Kapitel 5 ‚Diskussion' wird auf die einleitenden Fragestellungen und die Hypothese zurückgegriffen. Sie dienen als Gerüst bei der kritischen Betrachtung der verschiedenen Teile dieser Diplomarbeit. Es kann daraus geschlossen werden, dass die gesteckten Ziele erreicht sind. Ein theoretisches Grundgerüst zu den Visualisierungsoptionen ist erstellt. Für dessen Ausbaufähigkeit wird ebenfalls Platz eingeräumt. Auf diese theoretisch erarbeiteten Konzepte stützt sich ein selbständig implementierter Applikationsprototyp. Dabei kommen die zu Testzwecken generierten Daten von Eisschildmodellierungen zum Einsatz, die im NetCDF Format vorliegen. Durch die weite Verbreitung in der Klimamodellierung ist dieses Testgefäss realistisch. Zur Überprüfung der Applikation und der Visualisierungen wird eine qualitative Evaluation durchgeführt. Diese ermittelt ein Interesse an den erzeugbaren Darstellungen und an der Anwendung. Viele Verbesserungs-vorschläge und allgemeine Anregungen werden daraus gesammelt. Diese können in zukünftige Applikationsversionen einfliessen.

7.2 Erkenntnisse

Es erstaunt, wie ungenau der Begriff Unsicherheit im wissenschaftlichen Sprachgebrauch verwendet wird. Obwohl eine grosse Auswahl an verschiedenen Fachausdrücken existiert, die jeder für sich eine differenzierende Aussagekraft haben kann, werden die Begriffe doch oftmals als einander stellvertretend benutzt.

Die Bestimmung der Datenqualität und damit der Datenunsicherheiten ist keine einfache Aufgabe. Um Unsicherheiten zu quantifizieren, muss deren Herkunft und Grösse bekannt sein. Wenn dies allerdings klar definierbar ist, handelt es sich nicht mehr um Unsicherheit. Wie kann man die Ungenauigkeit eines Messinstruments messen, wenn dieses ungenau ist? Dies ist ein nicht immer lösbares Dilemma beim Umgang mit Unsicherheiten.

Es gibt verschiedene Arten von Unsicherheiten. In jedem Schritt des Datenverarbeitungsprozesses können Unsicherheiten (Messungenauigkeiten, Konzeptionsfehler, Verarbeitungsfehler, etc.) in die zu verarbeitenden Datensets gelangen. Mit aller Vor- und Umsicht ist es nicht möglich, alle Unsicherheitsquellen zu eliminieren. Die vielen verschiedenen Herkunftsarten der Unsicherheiten bedingen auch unterschiedliche Vorgehensweisen bei der Bestimmung der Unsicherheiten. Die Methoden zur Festlegung der Unsicherheitsgrössen können sich unterscheiden. Entscheidend ist es, ein Sensorium für die Omnipräsenz der Unsicherheit zu schaffen.

Die beschriebenen Grafikvariablen eignen sich in verschiedenem Ausmass zur Visualisierung von Unsicherheiten in Rasterdaten mittels bivariaten Darstellungen. Unter den klassischen sieben Bertin’schen Grafikvariablen sind zwei als besonders gut geeignet hervorzuheben. Dieses sind die Textur und der Farbwert. Ihre Eigenschaft, einen Einfluss auf die Helligkeit des dargestellten Objektes auszuüben, prädestiniert sie zur Vermittlung der in den Daten variierenden Unsicherheit. Mit abnehmender Helligkeit kann die Verschlechterung der Datenqualität intuitiv abgebildet werden. Der Farbton kann ebenfalls eingesetzt werden. Er beinhaltet aber den Nachteil, dass durch die Änderung der dargestellten Spektralfarbe keine eindeutige Beziehung zum Grad der Unsicherheit entsteht. Für die Grösse, die Form und die Orientierung gibt es theoretische Konzepte zum Einsatz in den Darstellungen. Deren praktischer Nutzen ist allerdings sehr beschränkt, weshalb sie als nicht geeignet eingestuft werden. Gänzlich ungeeignet ist die Platzierung. Von den besprochenen erweiterten Grafikvariablen sind zwei gut geeignet. Als Erstes sei die Farbsättigung genannt, die ebenfalls durch ihren differierenden Helligkeitswert einen direkten Bezug zu den Datenqualitäten zulässt. Als Zweites folgt die Klarheit, deren Transparenzkomponente hervorzuheben ist. Mit zunehmender Transparenz verschwindet die dargestellte Rasterzelle, die Unsicherheit nimmt zu. Das Muster kann ebenfalls verwendet werden, da es sehr stark der Textur gleicht. Dadurch, dass es aus mehreren Bertin’schen Grafikvariablen besteht, ist der Einsatz komplexer.

Es gibt viele unterschiedliche Visualisierungstechniken. Deshalb sollen an dieser Stelle nur die besonders vielversprechenden erwähnt und deren Vorteile herausgestrichen werden. Mittels Animation lassen sich viele Modellierungsvarianten auf schnelle und kompakte Weise miteinander vergleichen. Kann der Benutzer Einfluss auf die Abspielgeschwindigkeit und Reihenfolge der Varianten nehmen, wird die Flexibilität erhöht. Die 3-dimensionale Darstellung der Unsicherheiten als Datenqualitäts-Oberfläche ermöglicht eine intuitive Erfassung der Unsicherheitsquantität. Die statische Darstellung zweier univariater Karten gleichzeitig nebeneinander wird schon seit längerer Zeit genutzt. Ihre Stärke liegt in der klaren Unterscheidbarkeit von Datenwert und Datenqualität.

Die entworfene und implementiert Applikation zeigt das grosse Potential der Unsicherheits-Visualisierungen auf. Sie beweist, dass mit verhältnismässig geringem Entwicklungsaufwand sehr gute Resultate erzielbar sind. Der experimentelle Designansatz fördert viele konzeptuelle und strukturelle Verbesserungsmöglichkeiten. So verbessert eine hohe Modularität in der Klassenhierarchie die Erweiterbarkeit wie die Wartbarkeit der ganzen Applikation. Dieser Ansatz sollte noch konsequenter fortgesetzt werden. Der Applikationsprototyp soll Anregungen zu weiteren Entwicklungen in dieser Forschungsrichtung geben. Dem Autor sind keine weiteren Programme bekannt, die Daten auf diese spezielle Art darzustellen vermögen.

Die Evaluation vermittelt wertvolle Einsichten in die Benutzerperspektive. Grundsätzlich bringen die Testpersonen den implementierten Visualisierungen ein grosses Interesse entgegen. Es besteht ein Konsens, dass die Darstellungen einen Mehrwert schaffen und die Datenaufbereitung wie die Dateninterpretation ergänzen. Dies spricht für deren Nützlichkeit und zeigt, dass eine Nachfrage dafür

besteht. Die geäusserte Bereitschaft, solche Darstellungen gegebenenfalls einzusetzen, unterstützt diese Feststellung zusätzlich.

In der Bewertung der einzelnen Visualisierungen gehen die Meinungen auseinander. Rund die Hälfte der Testgruppe findet die Dithering-Darstellung am nützlichsten und klarsten, die andere Hälfte wählt dafür die Schattierungsdarstellung. Einig sind sich die Testpersonen darin, dass die komplizierteren Quadtree-Darstellungen weniger intuitiv sind und dadurch das Verstehen deren Inhaltes mehr Zeit in Anspruch nimmt. Sie beurteilen die Block- und Gitterdarstellungen nicht als schlecht, schätzen aber die einfacheren Visualisierungen mehr. Daraus lässt sich schliessen, dass bekanntere und eingängigere Darstellungen, die alltäglichere Symbolisierungen verwenden, bei der Entwicklung zu bevorzugen sind. Exotische Experimente werden eher kritisch hinterfragt. Ein weiterer Punkt ist die Anzahl an Visualisierungsoptionen, die mehrheitlich als genügend erachtet werden. Somit sind die Benutzer mit wenigen, eindeutigen Darstellungen zufrieden. Die Implementierung vieler Visualisierungsarten liefern einen geringen Zusatznutzen.

Die Befragung zeigt Verbesserungspotential bei der Vereinfachung der Applikationssteuerung. Die Testpersonen empfinden die Auswahl der Funktionen im Menu in Kombination mit dem GUI Design als zu umständlich.

Ein dialoggesteuertes Interaktionskonzept wird mehrfach empfohlen. Dies bietet verschiedene Vorteile. Die GUI wird übersichtlicher, da gleichzeitig weniger Steuerelement eingeblendet werden. Dies steigert insgesamt die Übersichtlichkeit der Applikation. Zusätzlich beansprucht die GUI dadurch weniger Bildschirmplatz, der so für Visualisierungsfenster genutzt werden kann.

Als letzter wichtiger Punkt soll die Erweiterung des Funktionsumfang angesprochen werden. Viele Testpersonen äussern sich positiv zu einem Ausbau der Funktionalitäten. Die Möglichkeit einer direkten Ansicht der numerischen Datenwerte wird begrüsst. Ebenso willkommen sind (einfache) statistische Funktionen. Diese Stellungnahmen verdeutlichen, dass sich die Benutzer einen Funktionsumfang über den reinen Visualisierungsteil hinaus wünschen.

Die gewonnen Erkenntnisse können als sinnvollen Beitrag zur Erforschung der Unsicherheitsvisualisierung herangezogen werden. Im Bereich der praktischen Implementierung wird mit der Entwicklung des Prototypen ein möglicher Ansatz präsentiert, der ein Versuch zum Schliessen der bestehenden Anwendungslücke darstellt.

7.3 Ausblick

Es werden Anregungen zu weiteren Forschungsarbeiten gegeben. Diese gliedern sich in die fünf Themenbereiche theoretische Betrachtungen, Begrifflichkeit, Datenformat, Applikation und Evaluationsinstrument.

7.3.1 Theoretische Betrachtungen

Vor allem die unterschiedlichen Visualisierungstechniken werden hier nur kurz erwähnt. Dabei lässt sich deren Visualisierungseignung nicht in vollem Umfang analysieren, sondern diese wurde im Hinblick auf die arbeitsspezifische Situation betrachtet. Es wird nur eine Auswahl an Techniken vorgestellt. Eine Betrachtung weiterer Techniken wäre sicher interessant. Dabei liesse sich auf der vorliegenden Arbeit aufbauen.

Der Fokus liegt bei einfach realisierbaren und leicht verständlichen Visualisierungen und Techniken. Es bleibt abzuklären, inwiefern der Einsatz komplexer Techniken einen Zusatznutzen zu generieren vermag. Die angesprochene visuelle Erweiterung in die dritte Dimension bietet einen zusätzlichen Freiheitsgrad (van der Wel et al. 1994: 325). Mit diesem lässt sich eine weitere Variable darstellen, oder die Unsicherheiten akzentuieren. Bei allen technischen Möglichkeiten dürfen die Aspekte des Nutzens und der Verständlichkeit nicht vergessen gehen. Die Visualisierungen dürfen nicht zu überladen werden.

Nur ganz am Rande zu Sprache gekommen ist die Übermittlung von Unsicherheitsdaten für sehbehinderte Menschen. Dabei ist die Übersetzung in Töne nur einer von mehreren geeigneten Ansätzen. Eine weitere Technik unterstützt die taktilen Fertigkeiten der Benutzer. Dazu müssen Karten im Hochdruckverfahren zum Einsatz gelangen, oder dreidimensionale Darstellungsbretter, deren Oberflächenbeschaffenheit verändert werden kann. Analog zum Computerbildschirm können dies Tastbretter sein, deren Darstellungen aus einzeln ansteuerbaren Stiften generiert werden.

In der Bearbeitung und Darstellung von Rasterdaten und Vektordaten ergeben sich Unterschiede. Durch adaptives Zoomen bieten Vektordaten in jeder Vergrösserung eine optimale Darstellung. Gewisse Grafikvariablen und Visualisierungstechniken können anders eingesetzt werden, als bei der Grundlage von Rasterdaten. Eine detaillierte Untersuchung, fokussierend auf Vektordaten würde einen sinnvollen Vergleich zu den verwendeten Rasterdaten geben.

7.3.2 Begrifflichkeit

Der Begriff Unsicherheit wird sehr allgemein verwendet und steht für viele Bedeutungen. Es existieren weitere Begriffe, die ähnlich oder gar synonym dazu eingesetzt werden. In den untersuchten Literaturquellen werden die möglichen Bezeichnungen sehr unterschiedlichen eingesetzt. Deshalb wäre die verstärkte Suche nach einer einheitlichen Terminologie sehr zu begrüssen. Es gibt dazu den Versuch von Atkinson (2002), der ein erster vielversprechender Schritt in diese Richtung ist. Es ist zu wünschen, dass diese Terminologie von Vielen übernommen und ergänzt wird.

7.3.3 Datenformat

In Folge dieser ungeklärten Begriffsdefinitionen entstand nie ein einheitliches Datenformat für die Beschreibung dieser Metadaten. Dabei wird nicht an ein physisches Datenformat gedacht, denn dieses wird schon aus marktstrategischen Gründen auch nie für die anderen Daten einheitlich werden. Die grösseren Softwareproduzenten behalten ihre proprietären Formate bei. Eine einheitliche Datenbeschreibung wäre dagegen sicher realisierbar. Die XML-Technik könnte einen Designansatz dazu bieten.

Das NetCDF Datenformat hat sich als sehr zuverlässig und schnell erwiesen. Durch seinen speziellen Aufbau, die schnellen Lese- und Schreibzugriffe sowie die dynamische Erweiterbarkeit bietet es grosses Potential, auch Metadaten damit zu speichern. Diese Zusatzinformation können als weitere Variable direkt mit den Modelldaten mitgeliefert werden. Dazu muss für jeden gewünschten Wert eine Variable angefügt werden. Somit kann für jeden Rasterwert nicht nur ein Durchschnittswert aller Modellierungen mitgeliefert werden. Es lassen sich auch die Standardabweichung, die Minimal- und Maximalwerte und weitere statistische Angaben zu jeder Rasterzelle speichern. Dies wäre eine Erweiterung der hier eingesetzten Unsicherheitswerte, die aus dem Vergleich zweier Modellierungsdateien simuliert werden. Mittels der offenen Dimension lassen sich beliebig viele solcher Modellierungszyklen in eine Datei schreiben.

7.3.4 Erweiterung der Applikation

Die GUI spielt in einer Applikation eine wichtige Rolle. Von ihr hängt die Einfachheit der Bedienung und damit verbunden die Qualität des Arbeitens mit der Applikation ab. Wie in den Kapiteln Evaluation und Diskussion beschrieben, gibt es eine breite Auswahl an Verbesserungsvorschlägen, die von potentiellen Benutzern vorgeschlagen werden. Zukünftige Applikationen, die die Visualisierung von Unsicherheiten ermöglichen, können auf diese Ideenbasis zurückgreifen. Somit verkürzt sich die Zeit für eine Vorevaluation, die einen entsprechenden Anforderungskatalog erst zu bestimmen hätte.

Ebenfalls bereits mehrfach angesprochen wurde die Erweiterung der Auswahl an Visualisierungen. Trotz der Rückmeldung der Testpersonen, die die Menge der Visualisierungen als genügend erachten, lassen sich weitere vielversprechende Darstellungen erdenken. Dabei sei auf das Grundlagenkapitel verwiesen, das einige Anregungen dazu beinhaltet, sowie auf das erste Unterkapitel dieses Kapitels.

Wird das NetCDF-Fomat wie weiter oben beschrieben verwendet, muss die Datenverarbeitung umgestellt oder erweitert werden. Es werden andere Zugriffsmethoden und Speicherverfahren notwendig. In einem nächsten Schritt kann die Applikation mit gänzlich anderen Datenformaten arbeiten. Um diese Erweiterungen zu integrieren, ist unter Umständen ein neues Applikationsdesign notwendig, das im Datenverarbeitungsbereich einen modularen Aufbau besitzt. Dabei ist die Klassenstruktur so zu wählen, dass nur die Schnittstellen zu den weiteren Datenformaten zu programmieren sind. Mit diesen Anpassungen kann aus der prototypischen Lösung eine allgemeinere entwickelt werden.

Eine grosse Aufgabe besteht in der Implementierung von Methoden, die Vektordaten zu verarbeiten. Dabei kann nur noch bedingt auf die vorliegende Applikation zurückgegriffen werden, da viele Berechnungen und Visualisierungen einen anderen Ansatz verlangen.

7.3.5 Evaluationsinstrument

Für die Bemessung zukünftiger Anwendungen wird ein Ausbau des Evaluationsinstrumentes empfohlen. Soll ein ausgereifteres Produkt als der vorgestellte Prototyp repräsentativ evaluiert werden, muss die Stichprobengrösse mindestens 30 Testpersonen betragen (Bahrenberg et al. 1990: 18). Dabei kann ein einheitlicheres, standardisierteres Verfahren angewendet werden, beispielsweise mittels Ausbau und Verfeinerung des verwendeten Fragebogens (siehe Anhang). Damit ist eine exaktere Normierung möglich und eine statistische Auswertungen der Testdaten sinnvoll. Je grösser diese Testreihe aufgebaut wird, desto präzisere Aussagen lassen sich für die untersuchte Software machen (Bahrenberg et al. 1990: 18). Dabei soll es sich allerdings um eine ausgereiftere Version handeln. Sonst ist der Testaufwand kaum gerechtfertigt. Zudem kann bei einem reinen schriftlichen Verfahren die Anzahl der gleichzeitig befragten Personen erhöht werden. Dabei ist zu überlegen, wie die Applikation vorgestellt wird. Zum einen ist eine geführte Präsentation ohne Einwirken der Testpersonen denkbar. Dabei kann gewährleistet werden, dass alle Testpersonen die identischen Funktionen und Visualisierungen in der gleichen Reihenfolge zu sehen bekommen. Zum anderen kann als Testeinstieg eine kurze Erklärung zur allgemeinen Bedienung und den Funktionen gegeben werden, um die Testpersonen anschliessend die Applikation selber erforschen zu lassen. In dieser Variante ist die Standardisierung weniger einfach zu bewerkstelligen. Entscheidend ist die Zeitersparnis bei beiden Varianten. Der Aufwand für qualitative Einzelgespräche ist in der beschriebenen Testphase nicht mehr gerechtfertigt und übersteigt schnell ein angemessenes Mass.

Literatur- und Quellenverzeichnis

Ahlqvist, O., Keukelaar, J., and Oukbir, K. (2000): Rough classification and accuracy assessment, International Journal of Geographical Information Systems 14(5): 475-496

Ahlqvist, O., Keukelaar, J., and Oukbir, K. (2003): Rough and fuzzy geographical data integration, International Journal of Geographical Information Systems 17(3): 223-234

Amrhein, C.G., (1991) Comments on Visualising Data Quality, in Beard, M.K., Buttenfield, B. P., and Clapham, S.B., NCGIA Research Initiative 7, Visualization of Data Quality, Scientific Report of the Specialist Meeting, 8-12 June, Castine, Maine

Atkinson, P. M., Foody, G. M. (2002): Uncertainty in Remote Sensing and GIS: Fundamentals, in Foody, G. M., Atkinson, P. M., (eds.), Uncertainty in Remote Sensing and GIS, Wiley, Chichester

Bahrenberg, G., Giese, E., Nipper, J. (1990): Statistische Methoden in der Geographie, Band 1, Univariate und bivariate Statistik, 3. überarbeitete Auflage, Teubner Studienbücher der Geografie, Stuttgart

Beard, M.K., Buttenfield, B. P., and Clapham, S.B. (1991): NCGIA Research Initiative 7, Visualization of Data Quality, Scientific Report of the Specialist Meeting, 8-12 June, Castine, Maine

Blok, C.A. (2000): Dynamic visualization in a developing framework for the representation of geographic data, Cybergeo (http://www.cybergeo.presse.fr/), 153, 17. Novembre 2000

Boluva, V. (2001): Farbmodelle, Ausarbeitung zum Blockseminar (WS 2001/2002), Fachhochschule Giessen-Friedberg, http://homepages.fh-giessen.de/~hg10013/Lehre/MMS/SS01_WS0102/Farbmodelle/ (letzter Zugriff 29.August 2005)

Burrough, P.A., McDonnell, R.A. (1998): Principles of Geographical Information Systems, Oxford University Press, Oxford

Buttenfield, B. P. (1991): Visualizing cartographic metadata, in Beard, M.K., Buttenfield, B. P., and Clapham, S.B., NCGIA Research Initiative 7, Visualization of Data Quality, Scientific Report of the Specialist Meeting, 8-12 June, Castine, Maine

Cartwright, W., Crampton, J., Gartner, G., Miller, S., Mitchell, K., Siekierska, E., and Wood, J. (2001): Geospatial Infomation Visualisation, User Interface Issues, Cartography and Geographic Information Systems 28(1): 45-60

Clarke, K.C., Teague, P.D., (1999): Cartographic symolization of uncertainty, in Shi, W., Goodchild, M.F., Fisher, P.F., Proceedings of The International Symposium on Spatial Data Quality, Hong Kong, 18-20 July 1999

CSEP (Computational Science Education Project) (1995): Introduction to Monte Carlo Methods, http://csep1.phy.ornl.gov/CSEP/MC/MC.html (letzter Zugriff 3. September 2005)

DCDSTF (Digital Cartographic Data Standards Task Force) (1988): The proposed standard for digital cartographic data, The american cartographer, 15(1): complete issue

Dutton, G. (1991): Probability Filtering for Fuzzy Features, in Beard, M.K., Buttenfield, B. P., and Clapham, S.B., NCGIA Research Initiative 7, Visualization of Data Quality, Scientific Report of the Specialist Meeting, 8-12 June, Castine, Maine

Ehlschlaeger, C.R. (2002): Representin Multiole Spatial Statistics in Generalized Elevation Uncertainty Models: Moving beyond the Variogramm, International Journal of Geographical Information Systems 16(3): 259-285

Evans, B.J. (1996): Dynamic display of spatial data-reliability: Does it benefit the map user?, Computer & Geosciences 23(4): 409-422

Fairbairn, D., Andrienko, G., Andrienko, N., Buziek, G., and Dykes, J. (2001): Representation and its Relationship with Cartographic Visualization, Cartography and Geographic Information Systems 28(1): 13-28

Fisher, P.F. (1994): Hearing the reliability in classified remotely sensed images, Cartography and Geographic Information Systems 21(1): 31-36.

Foody, G. M., Atkinson, P.M. (eds.) (2002): Uncertainty in Remote Sensing and GIS, Wiley, Chichester

Foody, G. M. (2003): Uncertainty, knowledge discovery and data mining in GIS, Pogress in Physical Geography 27(1): 113-121

Frühauf, K., Ludewig, J., und Sandmayr, H. (2004): Software Prüfung: Eine Anleitung zum Test und zur Inspektion (5. Auflage), vdf Hochschulverlag AG, Zürich

Geldermans, S., and Hoogenboom, M. (2001): GIS Visualization – The Killer Application?, Geoinformatics 2001(Oct): 6-9

Hagdorn, M.K.M. (2003): Reconstruction of the Past and Forecast of the Future European and British Ice Sheets and Associated Sea-Level Change, unpublished PhD thesis, University of Edinburgh

Hebeler, F., Purves, R.S., Hagdorn, M., and Hulton, N. (2005): Experiments on the sensitivity of modelled extent of Fennoscandian icesheets to representation of topography, Poster for EGU05

Ince, D. (1995): Software Quality Assurance: Student Introduction, McGraw-Hill Book Company Europe, London

Jones, C. (1997): Geographical Information Systems and Computer Cartography, Addison Wesley Longman Limited, Pearson Education Limited, Harlow

Kraak, M. J. (2000): About Maps, Cartography, Geovisualization and other Graphics, Geoinformatics 2000(Dec): 26-27

Krygier, J.B. (1994): Sound and Geographic Visualization, in MacEachrean, A. M., and Taylor, D. R. F., (eds.), Visualization in Modern Cartography, Pergamon, Oxford, 149-166.

Longley, P.A., Goodchild, M.F., Maguire, D.J., and Rhind, D.W. (2001): Geographic Information Systems and Science, Wiley, Chichester

Lynch, P. J., Horton, S. (2002): Web Style Guide (2nd Edition), Dithering, Yale University Press, http://www.webstyleguide.com/graphics/dither.html (letzter Zugriff 29.August 2005)

MacEachren, A. M. (1995): How maps work, representation, visualization and design, New York, The Guilford Press

MacEachren, A. M., and Kraak, M. J. (2001a): Research Challenges in Geovisualization, Cartography and Geographic Information Systems 28(1): 3-12

MacEachren, A. M., and Kraak, M. J. (2001b): Forschungsfragen der Geovisualisierung / Research Challenges in Geovisualization (summary), Kartographische Nachrichten 51(4): 204-207

Pierce, D.W. (2003): Ncview: a netCDF visual browser, University of California, Scripps Institution of Oceanography, San Diego, http://meteora.ucsd.edu/~pierce/ncview_home_page.html (letzter Zugriff 29.August 2005)

Samet, H. (1989): Applications of Spatial Data Structure: Computer Graphics, Image Processing, and GIS, Addison-Wesley Publishing Company, Reading (Massachusetts)

Schöning, R. (1996): Modellierung des potentiellen Waldbrandverhaltens mit einem Geographischen Informationssystem, unveröffentlichte Diplomarbeit, Geographisches Institut Universität Zürich

The Gimp Team (2005): The Gimp, GNU Image Manipulation Program, Homepage, http://www.gimp.org/ (letzter Zugriff 27.August 2005)

Trolltech (2005): Trolltech Homepage, Qt Overview, Oslo, http://www.trolltech.com/products/qt/index.html (letzter Zugriff 29.August 2005)

Unidata (2005): University Corporation for Athmospheric Research (UCAR): Unidata: NetCDF Data Format, Boulder, http://my.unidata.ucar.edu/content/software/netcdf/index.html (letzter Zugriff 29.August 2005)

van der Wel, F. J. M., Hootsmann, M. R., and Ormeling, F. (1994): Visualization of Data Quality, in MacEachrean, A. M., and Taylor, D. R. F., (eds.), Visualization in Modern Cartography, Pergamon, Oxford, 313-331

Wilmms, A. (1999): GoTo C++ Programmierung, Addison-Wesley-Longmann, München

Zadeh, L.A. (1965): Fuzzy sets, Journal of Information and Control, 8: 338-353

Zheng, T.Q., and Zhou, Q. (2001): Optimal spatial decision making using GIS: a prototyp of a real estate geographical information system (REGIS), International Journal of Geographical Information Systems 15(4): 307-321

Zhang, J., and Goodchild, M.F. (2002): Uncertainty in Geographical Information (Research Monographs in Geographic Information Systems), Taylor & Francis, London

A Fragebogen

Dokumentation

Hat ihnen die Anleitung geholfen?	O Ja O Nein O Weiss nicht
Benötigten sie nach dem Lesen der Anleitung noch weitere Hilfe zum Bedienen des Programmes?	O Ja O Nein O Weiss nicht
Wünschen sie ein grösseres Tutorial, das z.B. mehr Funktionen abdeckt?	O Ja O Nein O Weiss nicht

GUI

Ist die Bedienung des Programmes eher einfach und intuitiv oder eher kompliziert und umständlich?	O Eher einfach O Sowohl als auch O Eher kompliziert
Finden sie die Darstellung jeder Visualisierung in einem eigenen Fenster gut oder schlecht?	O Gut O Schlecht O Weiss nicht
Sollen alle Darstellungsoptionen in einem Fenster aufgelistet werden, oder ist die Aufteilung zwischen GUI und Visualisierungsfenstern sinnvoll?	O Zusammenfassen O Getrennt lassen O Getrennt, andere Aufteilung

Funktionen

Wünschen sie die Möglichkeit einer numerischen Ansicht der Daten zusätzlich zu den Visualisierungen?	O Ja O Nein O Weiss nicht
Sollten zusätzliche statistische Funktionen oder Darstellungen ins Programm aufgenommen werden?	O Ja O Nein O Weiss nicht

Visualisierungen

Ist das Konzept verständlich, Originaldaten und Differenzendaten in einer Darstellung zu kombinieren?	O Ja O Nein O Weiss nicht
Hat das vorgestellte Programm ihr Interesse gegenüber diesen kombinierten Darstellungen geweckt?	O Ja O Nein O Weiss nicht
Glauben sie, dass solche Darstellugen nützlich sind, also einen Mehrwert bieten?	O Ja O Nein O Weiss nicht
Können sie sich vorstellen, solche kombinierten Darstellungsvarianten für ihre Arbeit zu nutzen, falls das von ihnen verwendete Programm dies zulässt?	O Ja O Nein O Weiss nicht
Wie beurteilen sie die Anzahl der vorhandenen Differenzendarstellungen?	O Eher zu wenige O Genügend O Eher zu viele

B Sourcecode - Headerdateien

Bei der Programmierung mit C++ werden die Dateien üblicherweise in zwei Dateiteile aufgeteilt. Die Definitionen der Klasse, der Attribute und der Methoden stehen in den Header-Dateien (mit der Endung .h), die Ausführungen in den dazugehörenden Body-Teilen (mit der Endung .cpp). Hier werden alle Header-Dateien der beschriebenen Applikation aufgeführt. Der Nachdruck des gesamten Codes wäre zu umfangreich und deshalb nicht sinnvoll. Die Header-Dateien andereseits bieten einen guten Überblick über die Klassen und ihre Schnittstellen. Bei Interessse an den genauen Implementierungen der einzelnen Klassen und Methoden sei auf die beigelegte CD verwiesen. Da die Mainklasse ohne Header-Datei auskommt und für den Applikationsstart unbedingt erforderlich ist, wird sie als einzige Body-Datei aufgeführt.

Main

```
/*****************************************************************
@titel         : uncvismain.cpp
@descripton    : write working time stamps
@autor         : Sascha Oehler
@begin         : Mon, 2004-10-11
@last-modified : Tue, 2005-04-13
@version       : 6
*****************************************************************/

#include <qapplication.h>
#include "uncvisgui.h"

int main( int argc, char **argv )
{
    QApplication app( argc, argv );
    UncVisGui uv;
    uv.setCaption("UNCertainty VISualization tool - S.Oehler");
//    uv.showMaximized();
    uv.setGeometry(0,0,500,785);
    uv.show();
    return app.exec();

}
```

GUI

```
/*****************************************************************
@titel         : uncvisgui.h
@descripton    : gui-part. holds all interface elements and some
                 more (hidden) stuff like io-class...
@autor         : Sascha Oehler
@begin         : Mon, 2004-10-11
@last-modified : Wed, 2005-05-11
@version       : 20
*****************************************************************/

#ifndef UNCVISGUI_H
#define UNCVISGUI_H

#include <qmainwindow.h>
#include <qlistbox.h>
#include <qgroupbox.h>
#include <qlabel.h>
#include <qpushbutton.h>
#include <qlineedit.h>
```

```
#include <qslider.h>
#include <qmap.h>
#include <qcheckbox.h>

#include "uncvismessage.h"
#include "uncvisio.h"
#include "uncvisnetcdf.h"
#include "uncvisnetcdforig.h"
#include "uncvisnetcdfcalc.h"

class UncVisGui : public QMainWindow
{
    Q_OBJECT
public:
    UncVisGui(QWidget* parent=0, const char* name=0, WFlags f=0);
    ~UncVisGui();

private slots:
    ////// general gui part /////////
    void help();
    void about();
    void frameCoordinates();
    void changeZoomFactor(int factor);

    ////// netCDF gui part /////////
    void loadNcSettings();
    void addNcPathList();
    void clearNcPathList();
    void loadNcPathList();
    void fileFocusChanged();
    void addNcFrame();
    void addDifferenceFrame();
    void drawNcFrames();
    void playNcFramesPlus();
    void playNcFramesMinus();
    void playNcFrames(int direction);
    void changeNcFrameSettings();
    void changeAllNcFrameSettings();
    void removeNcFrame();
    void removeAllNcFrames();

private:
    /////// general used objects //////
    UncVisIO *uvIO;
    UncVisMessage *uvMessage;

    ////// netCDF gui part /////////
    typedef QMap<int, UncVisNetCDF*> NetCDFRasterMap; // stores all ihRaster
    NetCDFRasterMap myNetCDFRasterMap;

    QLabel *pathToNcFileLabel;
    QListBox *pathToNcFile;

    QLabel *displayedNcFilesLabel;
    QListBox *displayedNcFiles;
    QListBox *displayedNcFiles2;

    QLabel *variablesListBoxLabel;
    QListBox *variablesListBox;

    QGroupBox *zoomSliderGroup;
    QSlider *zoomSlider;

    QGroupBox *colorModeGroup;
    QListBox *colorMode;

    QGroupBox *timeSliceGroup;
    QLabel *timeSliceCountLabel;
    QLabel *timeSliceCount;
    QLabel *timeSliceMaxLabel;
    QLabel *timeSliceMaxCount;
    QSlider *timeSliceSlider;

    QGroupBox *limitGroup;
    QLabel *minZLimitLabel;
    QLineEdit *minZLimit;
    QLabel *maxZLimitLabel;
```

```
    QLineEdit *maxZLimit;
    QCheckBox *ownLimitsCBox;

    QGroupBox *playGroup;
    QPushButton *playNcFramesPlusB;    // Play Elements
    QPushButton *playNcFramesMinusB;    // Play Elements
    QLineEdit *playSteps;
    QLineEdit *playIncrement;
    QLineEdit *playBreaks;
    QLabel *playStepsLabel;
    QLabel *playIncrementLabel;
    QLabel *playBreaksLabel;
};

#endif
```

NetCDF

```
/*******************************************************************************
@titel         : uncvisnetcdf.h
@descripton    : basic class for netCDF frames. stores raster of netCDF
                 variable choosen from loaded file. Some basic gui components.
@autor         : Sascha Oehler
@begin         : Mon, 2004-11-08
@last-modified : Wed, 2004-05-11
@version       : 12
*******************************************************************************/

#ifndef UNCVISNETCDF_H
#define UNCVISNETCDF_H

#define YPOINTS 119  // TODO Change to flexible grid later...
#define XPOINTS 142  // TODO Change to flexible grid later...
#define YMENU 50     // The size of the 'menu-header', needed for frame-size calculation

#include <qframe.h>
#include <qstring.h>
#include <qslider.h>
#include <qlabel.h>
#include <qpushbutton.h>
//#include <qcheckbox.h>

#include <netcdfcpp.h>

class UncVisNetCDF : public QFrame
{
friend class UncVisPainter;
    Q_OBJECT

public:
    UncVisNetCDF(QWidget* parent=0, const char* name=0, WFlags f=WType_TopLevel);
    ~UncVisNetCDF();

    // Overloaded function for internal use...
    virtual void drawOneTimeSlice();
    // draw the actual Raster / timeSlice in zRaster-Array
    virtual void drawRaster();   // For CalcMix frames its a different one...

    QString getName() const;
    void setPathName(QString _pathName);
    QString getPathName() const;

    float getZMax() const;
    float getZMin() const;

    bool getOwnLimitsFlag() const;
    void setOwnLimitsFlag(bool _ownLimits);
    float getLimZMax() const;
    void setLimZmax(float _limZMax);
    float getLimZMin() const;
    void setLimZmin(float _limZMin);

    float getRasterPoint(int _x, int _y);
    void setRasterPoint(int _x, int _y, float _z);
```

```
    int getColorGradient() const;
    void setColorGradient(int _colorGradient);
    int getColorAlgorithm() const;
    void setColorAlgorithm(int _colorAlgorithm);

    int getZoomFactor() const;
    virtual void setZoomFactor(int _zoomFactor);

    QString getSelectedVariable() const;
    void setSelectedVariable(QString _selectedVar);

    int getTimeSliceActiv() const;
    virtual void setTimeSliceActiv(int _timeSliceActiv);
    virtual void setTimeSlider(int _maxValue, int _value);

    bool getIsChangeable() const;

public slots:
    virtual void setIsChangeable(bool _isChangeable);

protected:
    void setZMinMax();    // Sets the Min & Max value (zMin/zMax)

    float zRaster[YPOINTS][XPOINTS];  // Contains the raster data ('z-axis')
    int xPoints;   // amount of horizontal points
    int yPoints;   // amount of vertical points

    float zMax;    // maximum value found for z (the highest value)
    float zMin;    // minimum value found for z (the lowest value)
    bool ownLimits;  // Set true if other limits than zMin / zMax are desired
    float limZMax;   // limit-max value for drawing z
    float limZMin;   // limit-min value for drawing z

    int timeSlices;    // amount of time slices
                      // TODO this should load from netCDF, not inherit from GUI...
    int timeSliceActiv;    // which time slice is actually showed
    QString pathName;    // String containing the full path to the netCDF file
    QString name;    // String containing the 'id' of this netCDF-layer
    QString selectedVariable;    // String containing the variable displayed (from the netCDF
                                                                             file)
    int colorGradient;     // Flag for the color-gradient
    int colorAlgorithm;    // Flag for the color-algorithm
    int zoomFactor;    // Stores the zoom-factor multiplicator
    bool isChangeable;    // Is this frame changeable or not (similar to 'dynamic')
    QPushButton *isChangeablePB; // OushButton-control for the 'isChangeable' status

private:
};

#endif
```

NetCDFOrig

```
/*****************************************************************
@titel         : uncvisnetcdforig.h
@descripton    : stores/handles original netCDF frames
@autor         : Sascha Oehler
@begin         : Mon, 2004-12-13
@last-modified : Wed, 2005-05-11
@version       : 3
*****************************************************************/

#ifndef UNCVISNETCDFORIG_H
#define UNCVISNETCDFORIG_H

#include "uncvisnetcdf.h"

class UncVisNetCDFOrig : public UncVisNetCDF
{
    Q_OBJECT
public:
    UncVisNetCDFOrig(QWidget* parent=0, const char* name=0, Wflags
        f=WType_TopLevel);
```

```
    ~UncVisNetCDFOrig();

    // opens netCDF file and adds the data to the zRaster-Array
    void fillRasterWithNetCDFData();

public slots:
    // draws the frame and the raster, new raster calc etc...
    void drawOneTimeSlice();
    void setIsChangeable(bool _isChangeable);
    void setTimeSliceActiv(int _timeSliceActiv);
    void setTimeSlider(int _maxValue, int _value);

private:
    void init();  // initialize object
    QSlider *frameSlider;
    QPushButton *drawFrame;

};

#endif
```

NetCDFCalc

```
/*****************************************************************
@titel         : uncvisnetcdfcalc.h
@descripton    : stores frames calculated from orig. nc-files
@autor         : Sascha Oehler
@begin         : Mon, 2004-12-13
@last-modified : Wed, 2005-03-02
@version       : 5
*****************************************************************/

#ifndef UNCVISNETCDFCALC_H
#define UNCVISNETCDFCALC_H

#include <qcombobox.h>
#include <qstring.h>
#include <qcheckbox.h>

#include "uncvisnetcdf.h"

class UncVisNetCDFCalc : public UncVisNetCDF
{
    Q_OBJECT
public:
    UncVisNetCDFCalc(QWidget* parent=0, const char* name=0, WFlags f=WType_TopLevel);
    ~UncVisNetCDFCalc();

    // This is used for the difference frame (newer version)
    void setDifferenceLayers(UncVisNetCDF *_fromLayer, UncVisNetCDF *_toLayer);
    void fillRasterWithData();
    void setZoomFactor(int _zoomFactor);

public slots:
    // draws the frame and the raster, new raster calc etc...
    void drawOneTimeSlice();
    void drawComboRaster();

protected:
    // initialize object
    void init();

    UncVisNetCDF *fromNC;
    UncVisNetCDF *toNC;

private slots:
    void setVisualization(int type);

private:
    QComboBox *cBox;
    QCheckBox *diffAbsCB;
    int visualization;

    void setLimits();
```

```
};

#endif
```

Painter

```
/**************************************************************
@titel         : uncvispainter.h
@descripton    : this class displayes the raster on the screen
@autor         : Sascha Oehler
@begin         : Fri, 2004-11-19
@last-modified : Wed, 2005-04-27
@version       : 8
**************************************************************/

#ifndef UNCVISPAINTER_H
#define UNCVISPAINTER_H

#define BOTTOM YPOINTS+24

#include <qpainter.h>
//#include <qstring.h>
#include <qfont.h>

#include "uncvisnetcdf.h"

class UncVisPainter : public QPainter
{
//    Q_OBJECT
public:
    UncVisPainter( const QPaintDevice * pd, bool unclipped = FALSE );
    ~UncVisPainter();

    void drawRaster(UncVisNetCDF *_zLayer, QString titel);
  // Sub-methods for different drawing routines...
  // TODO change this, bad coding - redundant see manual for ideas
    void drawRasterBW(UncVisNetCDF *_zLayer);
    void drawRasterHue(UncVisNetCDF *_zLayer);
    void drawRasterSaturation(UncVisNetCDF *_zLayer);
    void drawRasterValue(UncVisNetCDF *_zLayer);

  // TODO Experimental stage - could have some general ideas - see manual for notes
    void drawRasterMix(UncVisNetCDF *_diffLayer, UncVisNetCDF *_fromLayer);
    void drawRasterDither(UncVisNetCDF *_diffLayer, UncVisNetCDF *_fromLayer);
    void drawRasterDouble(UncVisNetCDF *_fromLayer, UncVisNetCDF *_toLayer, int
                  version=0);
    void drawRasterQuadtree(UncVisNetCDF *_diffLayer, UncVisNetCDF *_fromLayer, int
                  version=0);

public slots:

private:
    float limZMax1;
    float limZMin1;
    float zMax1;
    float zMin1;
    float limZMax2;
    float limZMin2;
    float zMax2;
    float zMin2;
    int colorAlgorithm;
    int zoom;
    int legendBottom;
    int legendLeft;
    int legendColor;

    void settings(UncVisNetCDF *_zLayer1, UncVisNetCDF *_zLayer2);
//    void drawLegend();
    void drawLegendHelp(int layers);
    void legendDescription(QString titel, int layerNo, int position);
    void quadtree1(UncVisNetCDF *_diffLayer, UncVisNetCDF *_fromLayer, int xFrom, int xTo, int
                  yFrom, int yTo, int xLoopPos, int yLoopPos);
    void quadtree2(UncVisNetCDF *_diffLayer, UncVisNetCDF *_fromLayer, int xFrom, int xTo, int
                  yFrom, int yTo, int xLoopPos, int yLoopPos);
```

```
    void quadtree3(UncVisNetCDF *_diffLayer, UncVisNetCDF *_fromLayer, int xFrom, int xTo, int
                      yFrom, int yTo, int xLoopPos, int yLoopPos);
    void quadtree4(UncVisNetCDF *_diffLayer, UncVisNetCDF *_fromLayer, int xFrom, int xTo, int
                      yFrom, int yTo, int xLoopPos, int yLoopPos);
    void helpRaster(int maxQuadTree, int xBlocks, int yBlocks);

};

#endif
```

IO

```
/****************************************************************************
@titel          : uncvisio.h
@descripton     : this is the loader class, used for all file access methods
@autor          : Sascha Oehler
@begin          : Mon, 2004-10-04
@last-modified : Fri, 2005-04-29
@version        : 11
****************************************************************************/

#ifndef UNCVISIO_H
#define UNCVISIO_H

#include <qwidget.h>
#include <qstring.h>
#include <qstringlist.h>
#include <qlistbox.h>
//#include <qcombobox.h>
#include <qlineedit.h>
#include <qlabel.h>
#include <qslider.h>

#include <netcdfcpp.h>

class UncVisIO : public QWidget
{
    Q_OBJECT
public:
    UncVisIO(QWidget *parent=0, const char *name=0) : QWidget(parent, name) {};
    UncVisIO::~UncVisIO(){};

public slots:
    // opens a file with one timestep from the ice-sheet calculations variable ih
    void initNetCDFlistBox(QListBox &_netCDFlistBox);
    void initNetCDFsettings(QListBox &_netCDFlistBox, QLineEdit &_minZLimit, QlineEdit
                              &_maxZLimit );
    void netCDFfileChoosed(QListBox &_netCDFvarBox, QString path, QLabel &_countMax, QSlider
                             &_timeSliceSlider);
    void addNetCDFfiles(QListBox &_netCDFlistBox);
};

#endif
```

Messages

```
/*****************************************************************
@titel          : uncvismessage.h
@descripton     : class for all messages
@autor          : Sascha Oehler
@begin          : Thu, 2005-04-28
@last-modified :
@version        : 1
*****************************************************************/

#ifndef UNCVISMESSAGE_H
#define UNCVISMESSAGE_H

#include <qwidget.h>

class UncVisMessage : public QWidget
```

```
{
    Q_OBJECT
public:
    UncVisMessage(QWidget *parent=0, const char *name=0);
    ~UncVisMessage();

    void showMessage(int uid);
    void showMessage(QWidget *parent, int uid);
};

#endif
```

C Installation

Benötigte Software

Zum Kompilieren der Applikation muss das c++ Framework Qt (ab Version 3 oder höher) von Trolltech (http://www.trolltech.com/developer/index.html) installiert sein. Ausserdem wird die netCDF library (http://my.unidata.ucar.edu/content/software/netcdf/index.html) verwendet. Nach der Installation wird die Applikation folgendermassen installiert.

Kurze Version

- `tar -xf archivname.tar`
- `qmake`
- `make`
- `./uncvis.version.number`

Lange Version

Als erstes muss die Archiv-Datei im gewünschten Ordner entpackt werden. In Linux geschieht das mittels folgendem Befehl:

```
tar -xf archivname.tar
```

Folgende Dateien sollten ins Verzeichnis entpackt worden sein:

- Benutzer-Handbuch.html (Diese Datei lesen sie gerade)
- docu/ (Verzeichnis mit vielen .png Bilder und ein paar .html Seiten)
- settings.txt
- uncvis.pro
- uncvisgui.cpp
- uncvisgui.h
- uncvisio.cpp
- uncvisio.h
- uncvismain.cpp
- uncvismessage.cpp
- uncvismessage.h
- uncvisnetcdf.cpp
- uncvisnetcdf.h
- uncvisnetcdfcalc.cpp
- uncvisnetcdfcalc.h
- uncvisnetcdforig.cpp

- uncvisnetcdforig.h
- uncvispainter.cpp
- uncvispainter.h

Nun kann das Programm kompiliert werden. Dazu muss ein Makefile erzeugt werden. Dies geht ganz einfach, indem der Befehl

```
qmake
```

eingegeben wird. Damit startet man ein Programm gleichen Names, welches von Trolltech in ihrem qT Paket mitgeliefert wird. Diese Tool interpretiert eine .pro Datei (im ascii-Format, hier uncvis.pro) aus dem aktuellen Verzeichnis und erstellt daraus das erforderliche Makefile.

Nun fehlt nur noch das eigentliche Kompilieren. Das geht nun wie immer mittels Eingabe von

```
make
```

Als Resultat erhält man eine ausführbare Datei mit dem Namen uncvis.versions.nummer.